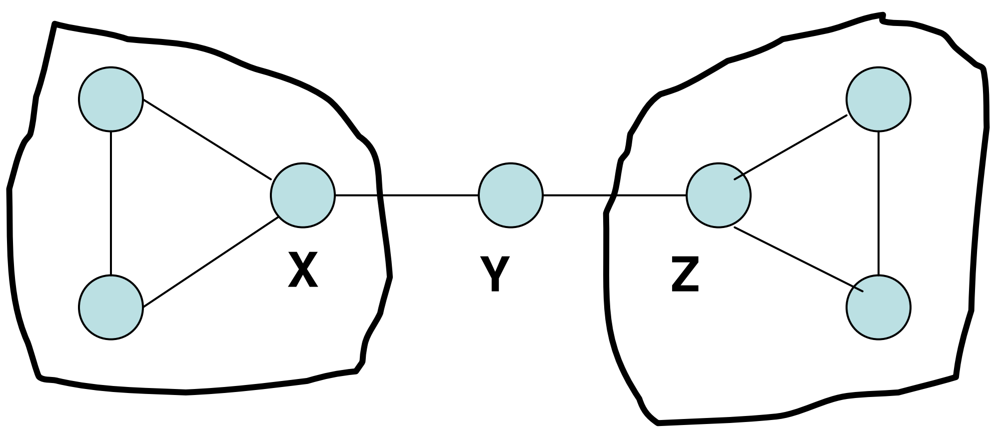
Centrality

Dr. Natarajan Meghanathan
Associate Professor of Computer Science
Jackson State University, Jackson, MS
E-mail: natarajan.meghanathan@jsums.edu

Centrality

- Tells us which nodes are important in a network (instead of just looking at the popularity of nodes)
  - How influential a person is within a social network
  - Which genes play a crucial role in regulating systems and processes
  - Infrastructure networks: if the node is removed, it would critically impede the functioning of the network.

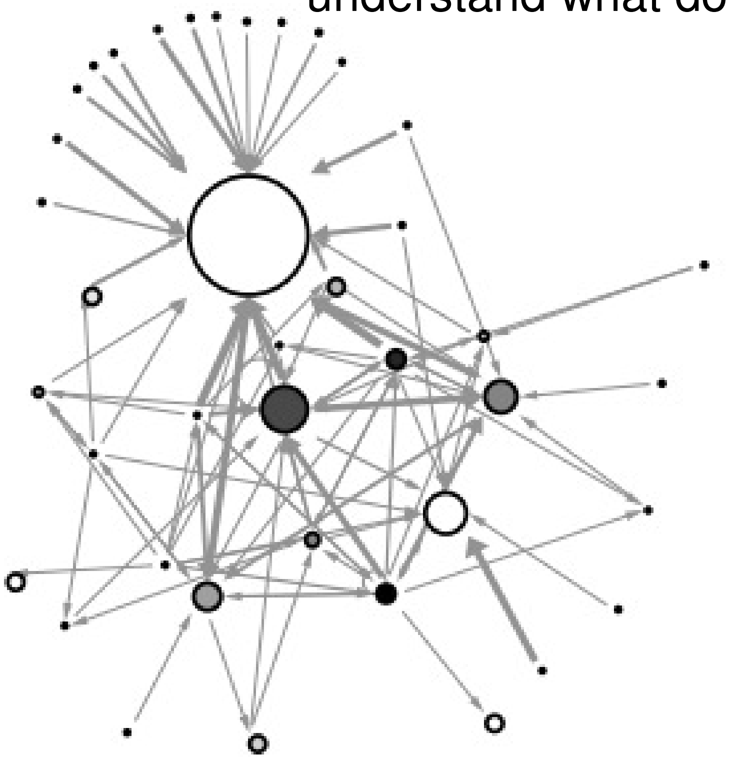


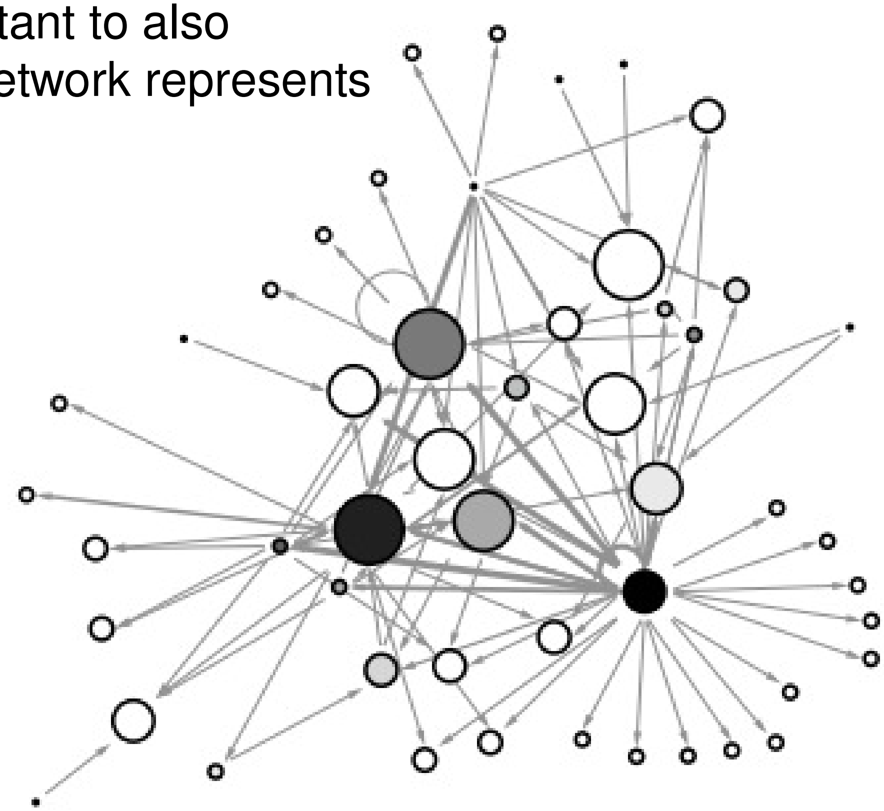
Nodes X and Z have higher Degree

Node Y is more central from the point of view of
Betweenness – to reach from one end to the other
Closeness – can reach every other vertex in the fewest number of hops

Example of a real-world trading network

Besides, topology it is important to also understand what does the network represents

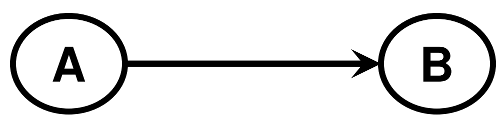




One node buying from many others
High In-Centralization

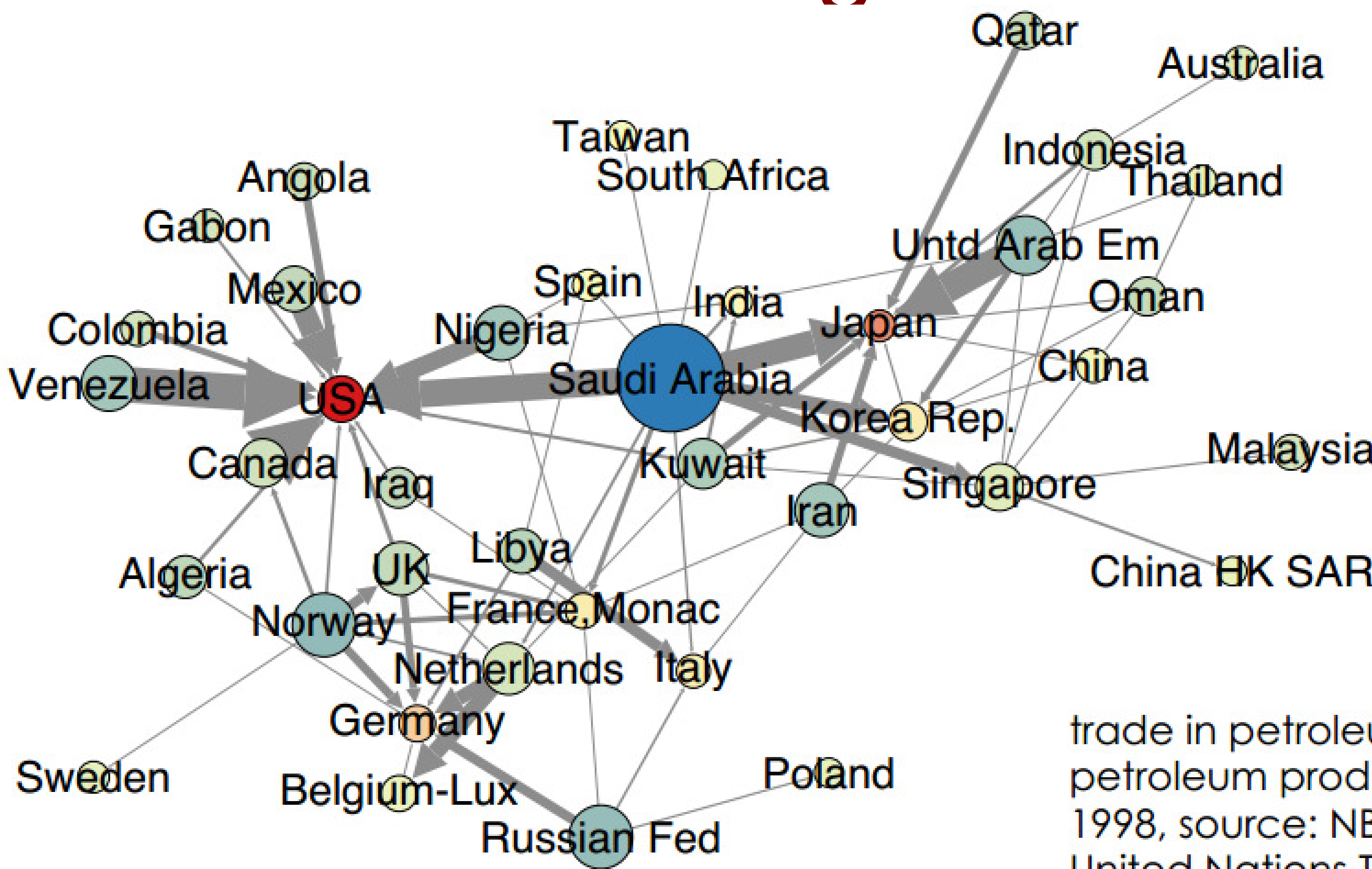
Buying is more evenly distributed

- Size of a node (buying): in-degree
- Darkness of the color of a node (selling): out-degree



Node A selling to Node B (Node B buying from Node A)

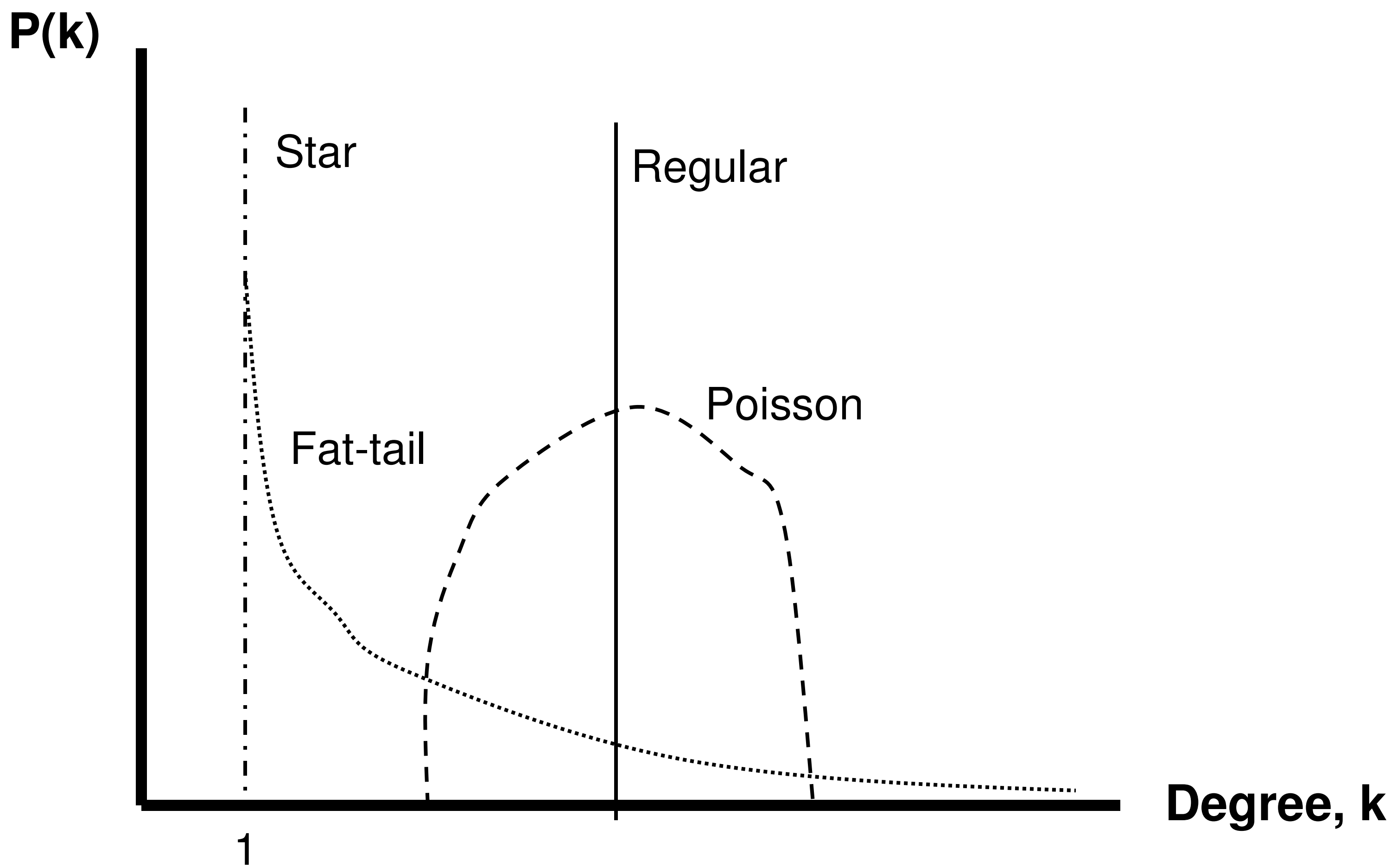
Out-degree



trade in petroleum an petroleum products, 1998, source: NBER-United Nations Trade Data

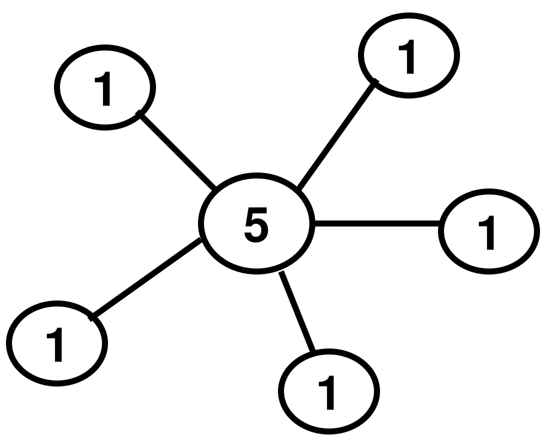
- The thickness of an edge could indicate the weight (e.g., amount of oil exported), the radius of the node circles could indicate the number of exports; node color – ratio of the number of exports to imports.

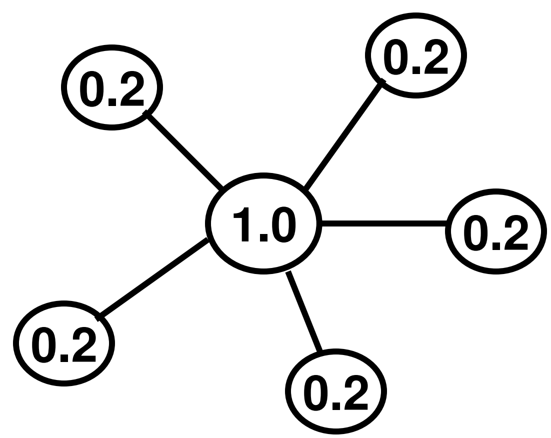
Typical Degree Distribution of Common Networks



Undirected Degree Centrality

- <u>Degree centrality for an individual node</u>
  - Just count the numbers
  - Sometimes, we normalize: divide the degree by N-1, where N - # nodes
    - (not suited for larger networks)





Centralization: Skewness in Distribution

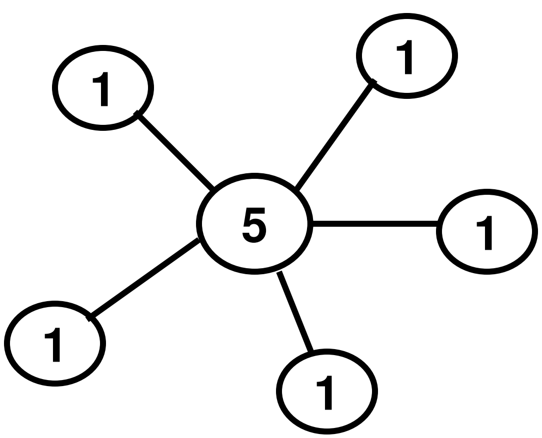
- Captures the variations in the centrality scores among the nodes in a network.

$$C_X^{Network} = \frac{\sum_{i=1}^{N} \left[C_X(n^*) - C_X(i) \right]}{\left[(N-1)(N-2) \right]}$$

X – Degree; Closeness; Betweenness

n* - max. centrality score

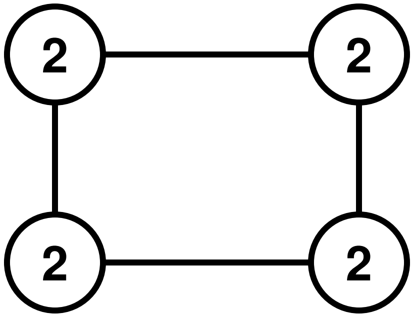
- <u>Example for Degree Centralization of a Network</u>




```
[ (5 – 5) + (5 – 1) * 5 ]         20
----------------------------  =  ----------  =  1.0
     [(6-1)(6-2)]                 5 * 4
```

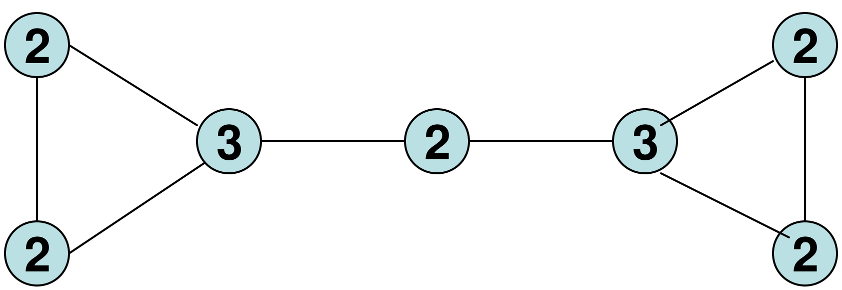
If the Degree Centralization of a Network is 1.0, it indicates a hub and spoke network

Degree Centralization of a Network



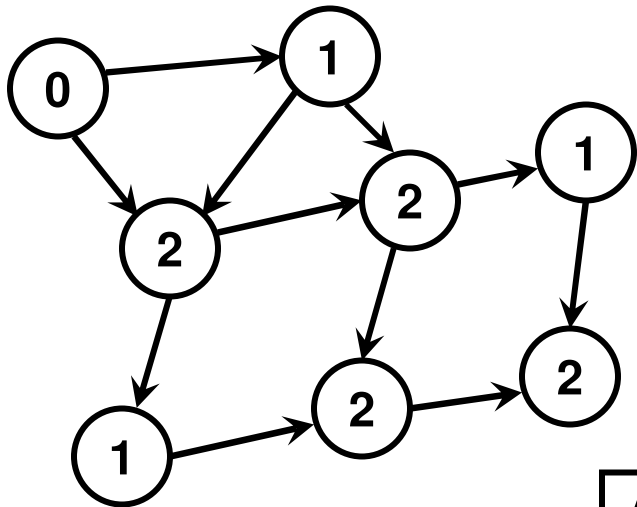
Degree Centralization of the Network = 0.0
- All nodes have the same degree

$$\frac{[\ (2-2)*2 + (2-1)*2\ ]}{[(4-1)(4-2)]} = \frac{2}{3*2} = 0.33$$



$$\frac{[\ (3-3)*2 + (3-2)*5\ ]}{[(7-1)(7-2)]} = \frac{5}{6*5} = 0.167$$

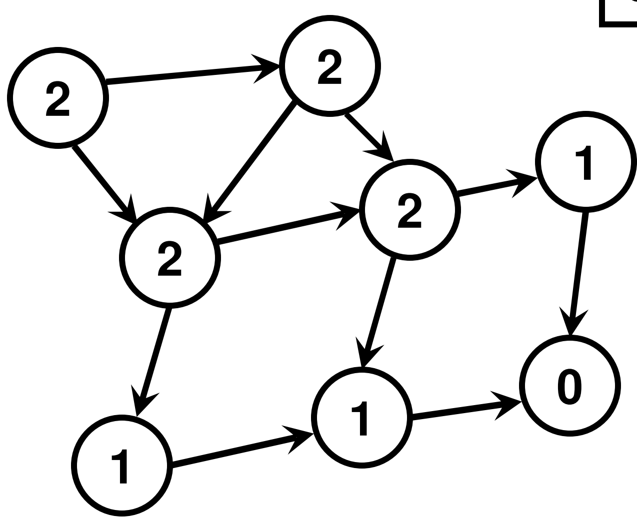
Degree Centralization: Directed Graphs



<u>In-Centralization</u>

$$\frac{[(2-2)*4 + (2-1)*3 + (2-0)*1]}{[(8-1)(8-2)]} = \frac{3 + 2}{7*6} = 0.119$$

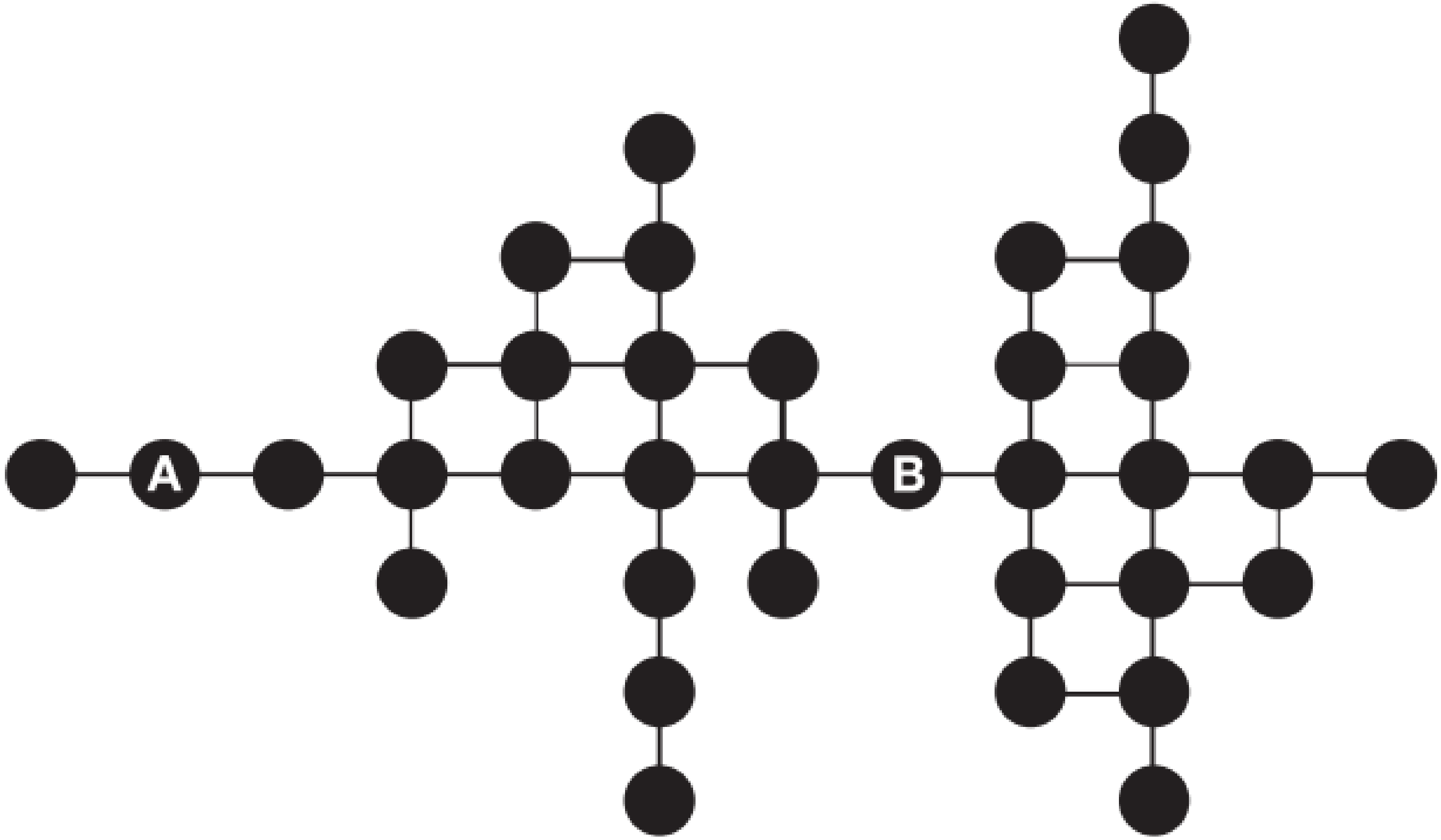
As the sum of the in-degrees of the vertices should be equal to the sum of the out-degrees of the vertices, in-centralization = out-centralization



<u>Out-Centralization</u>

$$\frac{[(2-2)*4 + (2-1)*3 + (2-0)*1]}{[(8-1)(8-2)]} = \frac{3 + 2}{7*6} = 0.119$$

Problem with Degree Centrality



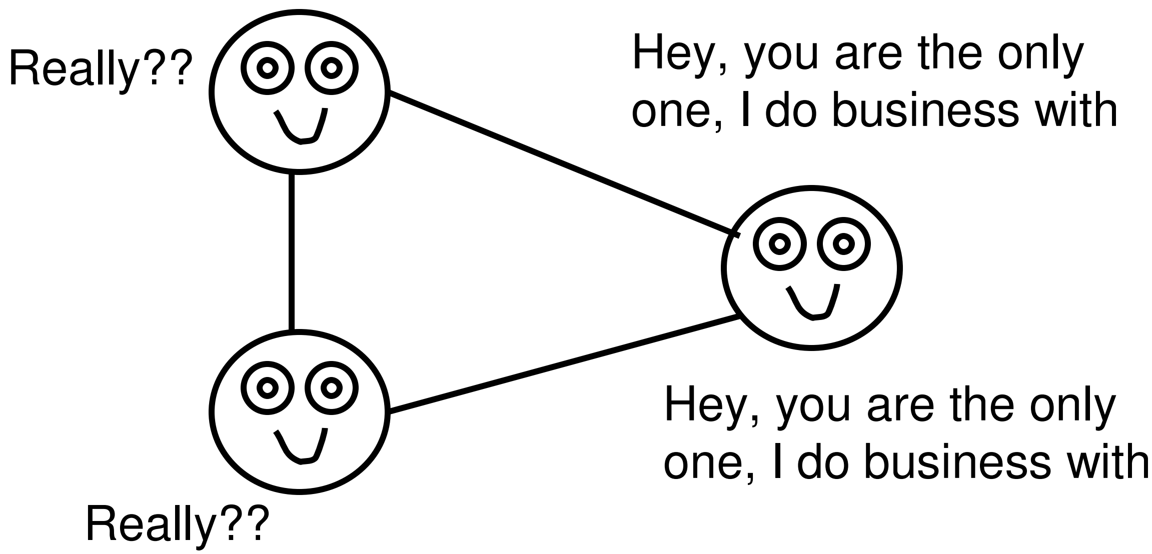
Both nodes A and B have the same degree (2)
But, node B is much more important to the functioning of the network than node A

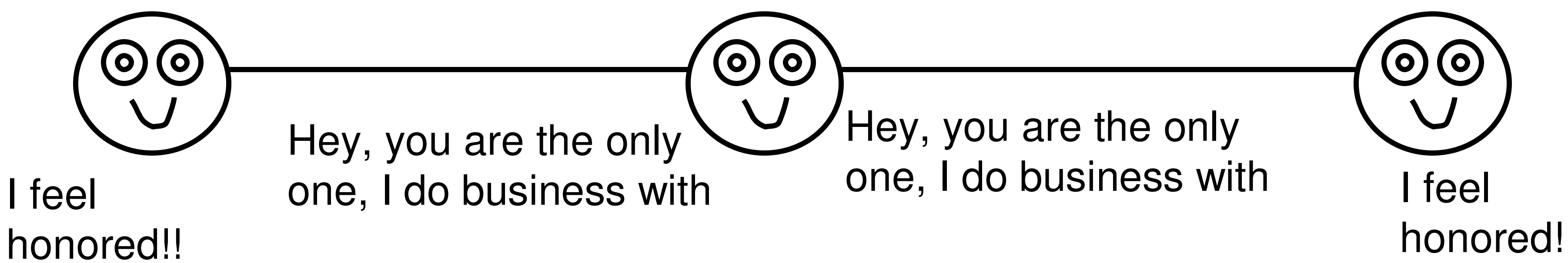
Centrality: Four Primary Ways

- Degree – connectedness
- Closeness, Decay – ease of reaching other nodes
- Betweenness – role as an intermediary, connector
- Influence, Prestige, Eigenvectors
  - “Not what you know, but who you know”

- Lots of different measures that capture different aspects of information
  - Needs to be used depending on the context
- There is not one that is the best

Betweenness: Brokerage





Betweenness: Brokerage

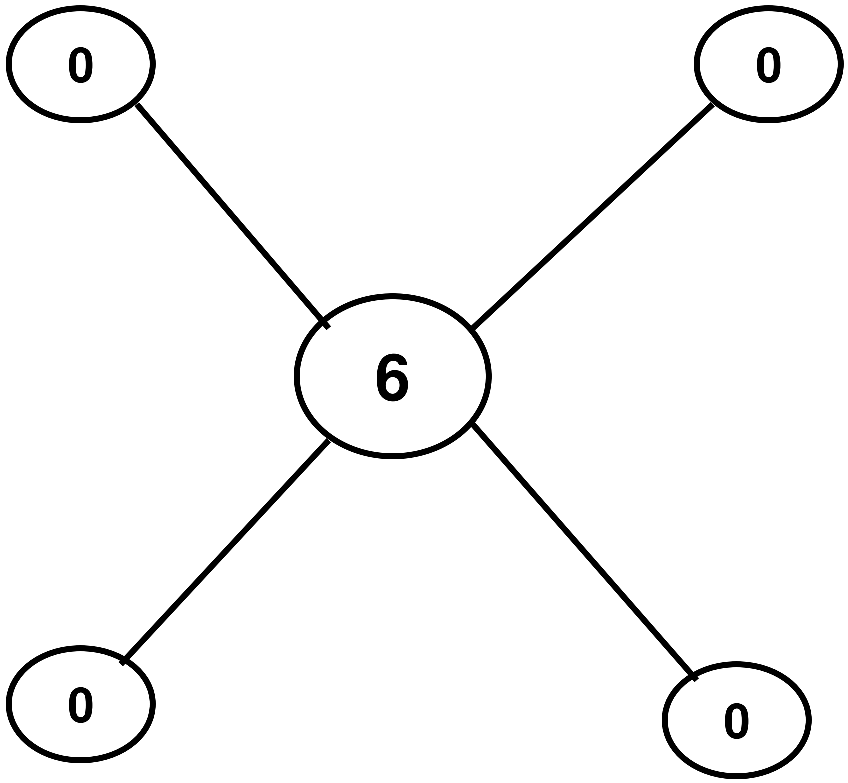
- A measure of how significant a node is in facilitating communication between the other nodes in the network (on the shortest path – min. # hops).
  - If several nodes in the network can communicate only by going through node X, then node X is said to have high Betweenness Centrality.

A measure of the fraction of the Shortest paths a node lies on

$$C_B(i) = \sum_{j<k} sp_{jk}(i)$$

$sp_{jk}(i)$ - # shortest paths between vertices j and k that go through vertex i

Betweenness on Toy Networks (1)



There are 4*3/2 = 6 node pairs

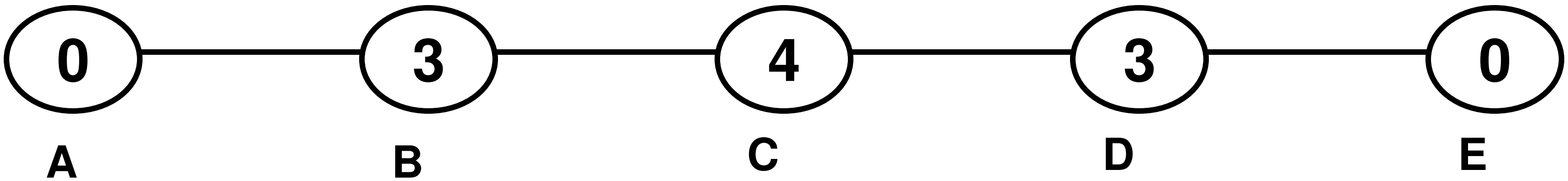
The central node lies on the shortest path for each node pair

There is only one shortest path between each node pair

Hence, the Betweenness of the central node is 6.

$$\text{Betweenness Centralization} = \frac{[\,(6-6)*1 + (6-0)*4\,]}{[\,(5-1)*(5-2)\,]} = \frac{24}{4*3} = 2.0$$

Betweenness on Toy Networks (2)

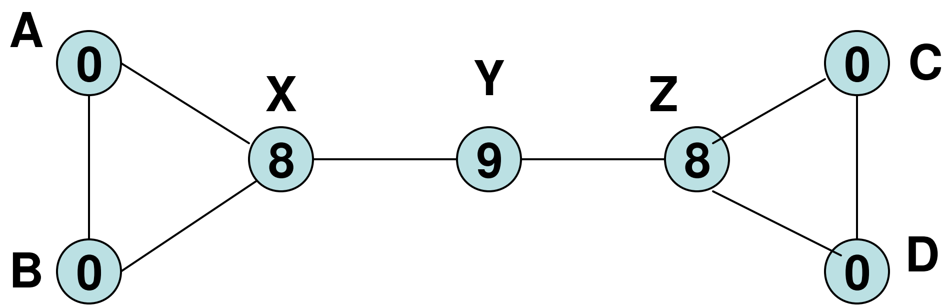


B is on the shortest path from A to C, A to D, A to E. Likewise,
D is on the shortest path from E to C, E to B, E to A.
C is on the shortest path from A to D, A to E
B to D, B to E

There is only one shortest path between any two vertices.

$$\text{Betweenness Centralization} = \frac{[\,(4 - 4)*1 + (4 - 3)*2 + (4 - 0)*2\,]}{[\,(5-1) * (5-2)\,]} = \frac{10}{4*3} = 0.83$$

Betweenness on Toy Networks (3)

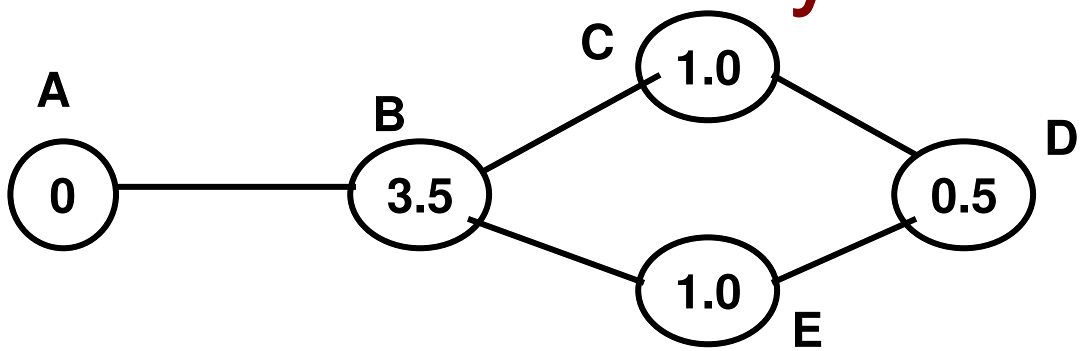


X is on the shortest path from A to C, A to D, B to C, B to D.
A to Y, A to Z, B to Y and B to Z
Z is on the shortest path from C to A, C to B, D to A, D to B
C to X, C to Y, D to X and D to Y.
Y is on the shortest path from A to C, A to D, B to C, B to D.
A to Y, A to Z, B to Y and B to Z; X to Y

There is only one shortest path between any two vertices.

$$\text{Betweenness Centralization} = \frac{[(9-9)*1 + (9-8)*2 + (9-0)*4]}{[(7-1)*(7-2)]} = \frac{38}{30} = 1.27$$

Betweenness on Toy Networks (4)



B: B is on the shortest path from A to C, A to D and A to E
(only one shortest path exist between these pairs of vertices)
For the pair C-E, there exists a shortest path through B and another one through D. B does not lie on the shortest path of any other pairs.
Betweenness of B = (1/1) + (1/1) + (1/1) + (1/2) = 3.5.

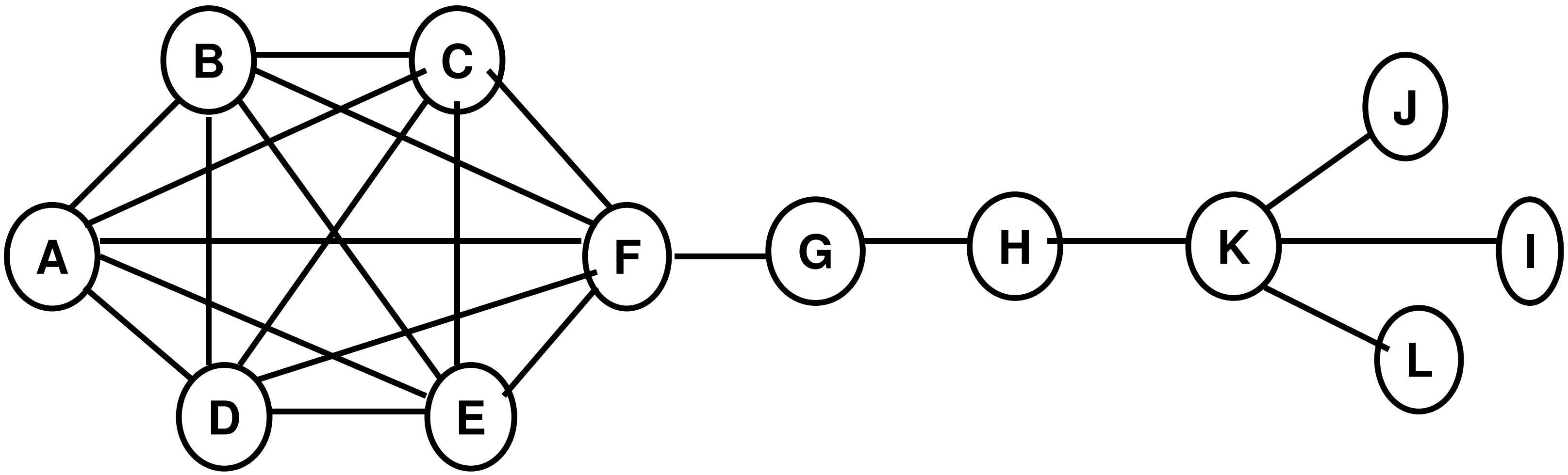
C: C lies on the shortest path from A to D, B to D. But there is also another shortest path from A to D (through E) and from B to D (through E)
Betweenness of C = ½ (A to D) + ½ (B to D) = 1.0

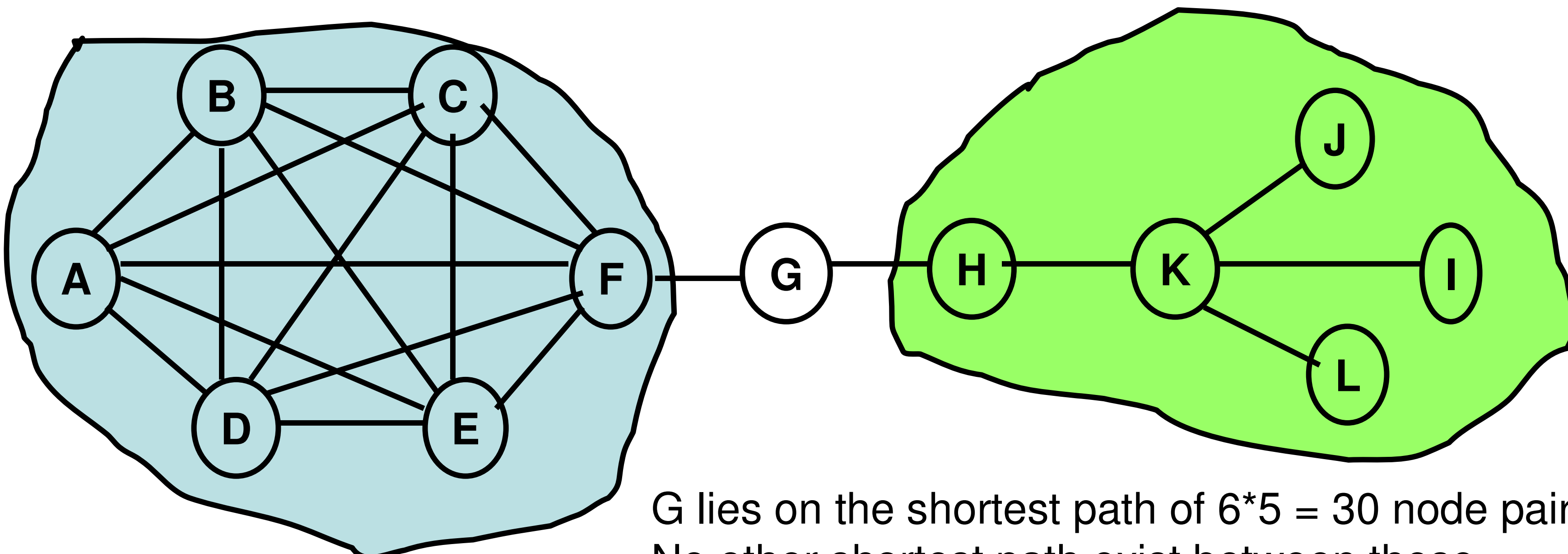
E: Like C, the Betweeness of E = ½ (A to D) + ½ (B to D) = 1.0

D: Betweenness of D = ½ (C to E) = 0.5.

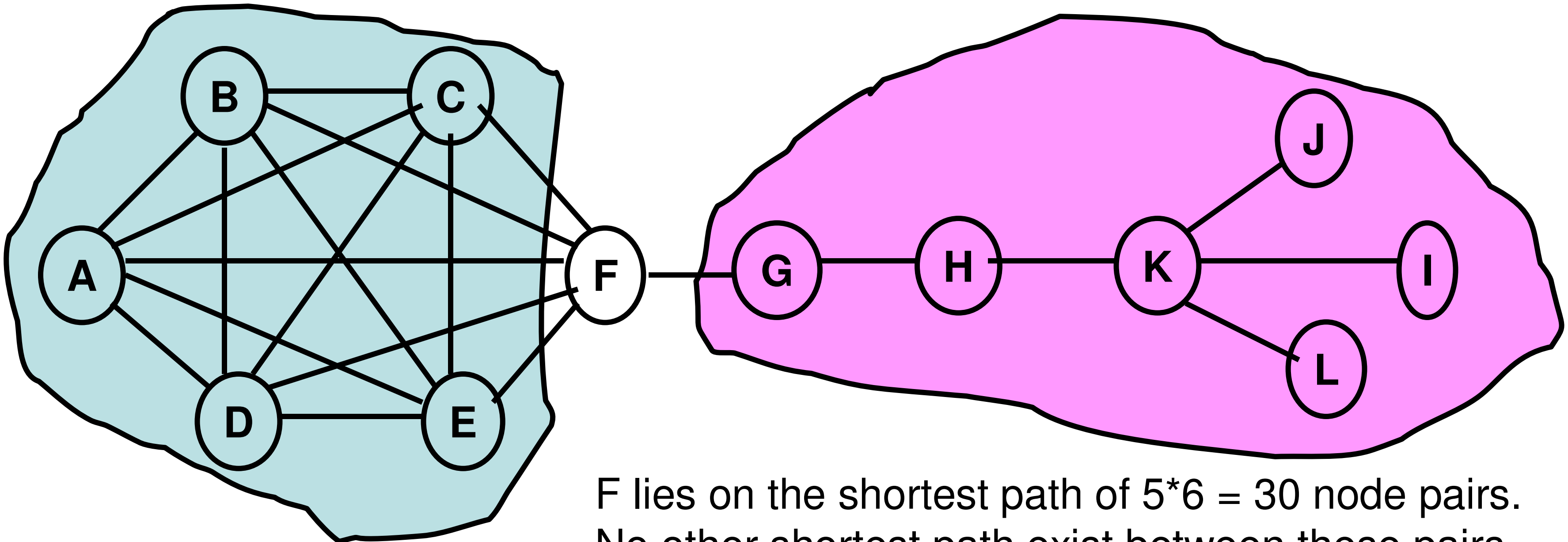
$$\text{Betweenness Centralization} = \frac{[(3.5 - 3.5)*1 + (3.5 - 1.0)*2 + (3.5 - 0.5)*1 + (3.5 - 0)*1]}{[(5-1)(5-2)]} = 0.958$$

Betweenness on Toy Examples (5)

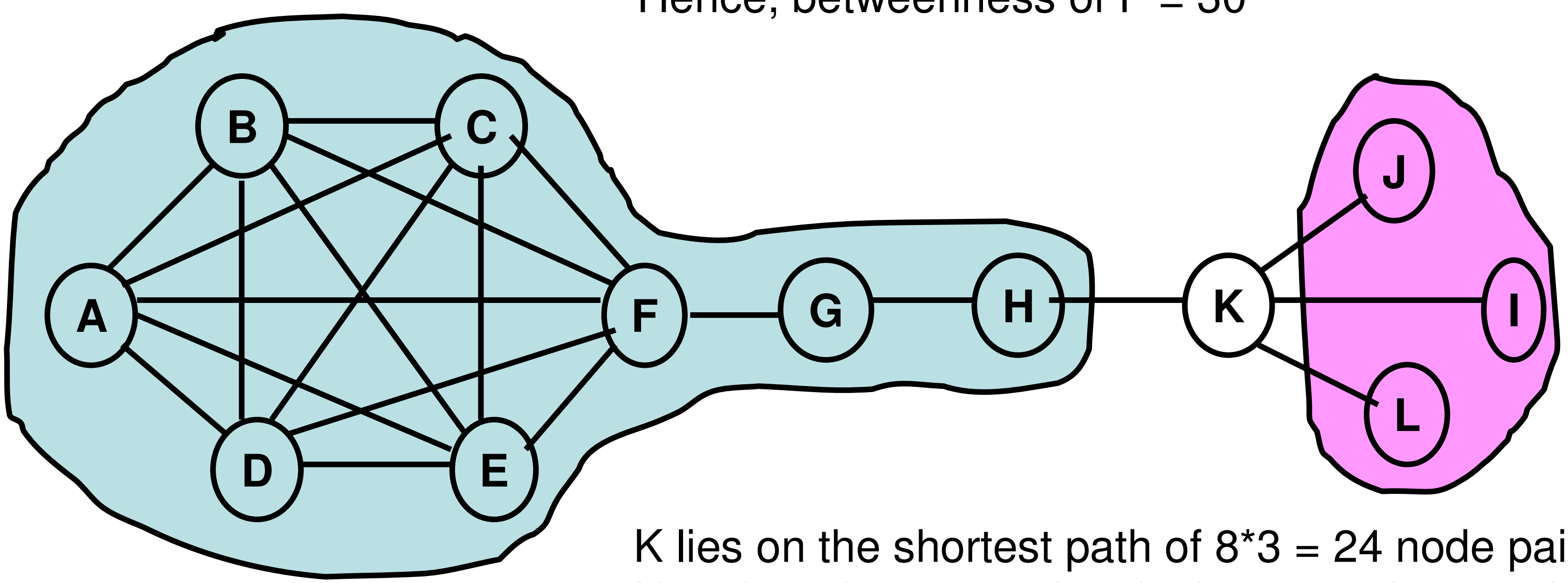




G lies on the shortest path of 6*5 = 30 node pairs.
No other shortest path exist between these node pairs. Hence, betweenness of G = 30

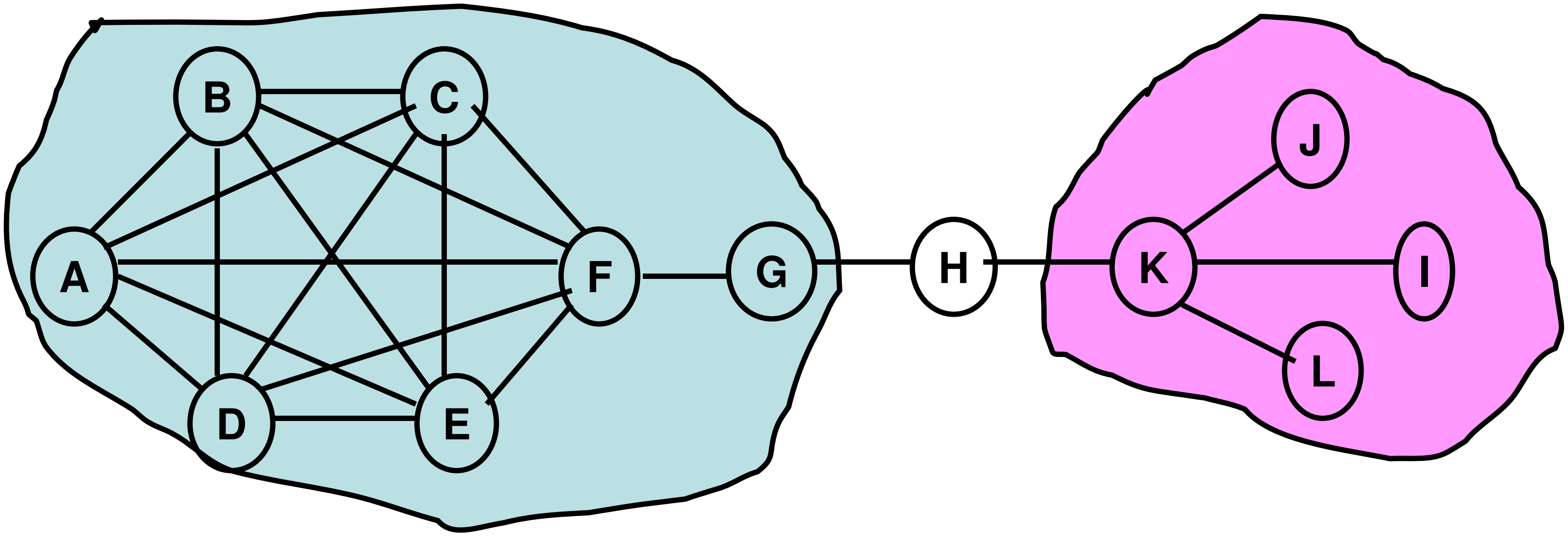


F lies on the shortest path of 5*6 = 30 node pairs.
No other shortest path exist between these pairs.
Hence, betweenness of F = 30



K lies on the shortest path of 8*3 = 24 node pairs.
No other shortest path exist between these pairs.
Hence, betweenness of K = 24

Betweenness on Toy Examples (5)



H lies on the shortest path of 7*4 = 28 node pairs. No other shortest path exist between these node pairs. Hence, betweenness of H = 28

Nodes A, B, C, D and E have high degree; but the betweenness for each is 0. Likewise, the betweenness of I, J and L is also 0.

$$\text{Betweenness Centrality} = \frac{[\,(30 - 30)*2 + (30 - 24)*1 + (30 - 28)*1 + (30 - 0)*8\,]}{[\,(12-1)(12-2)\,]} = 2.25$$

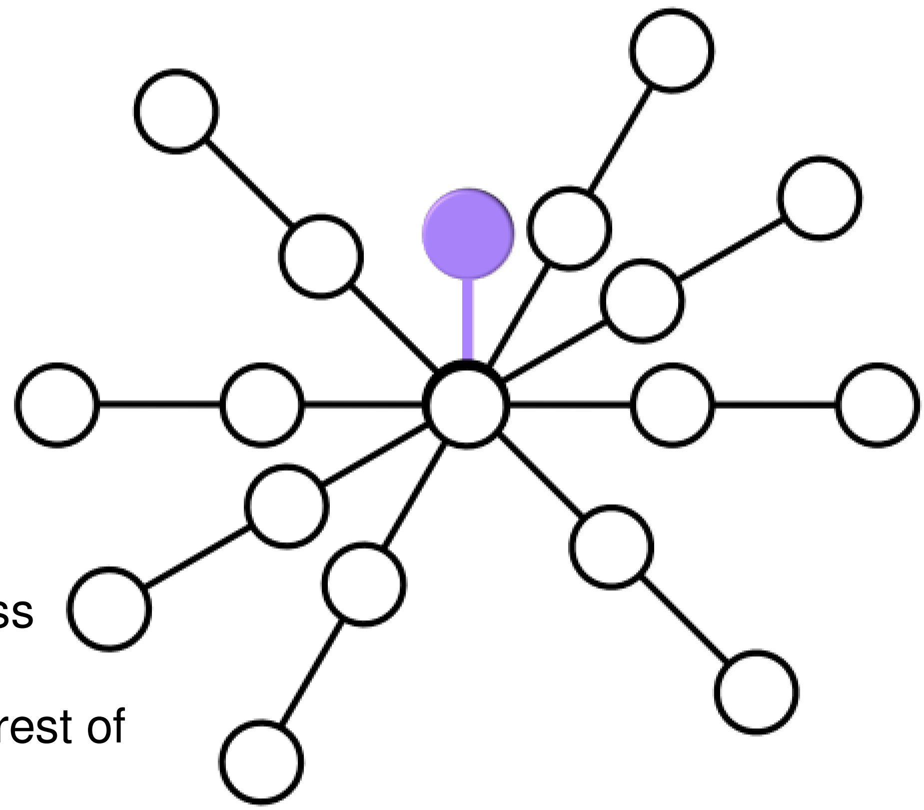
Closeness

- A measure of how far away is the rest of the network from a certain node.
  - Based on the length of the average shortest path between a node and all other nodes.

If the network is one single connected component

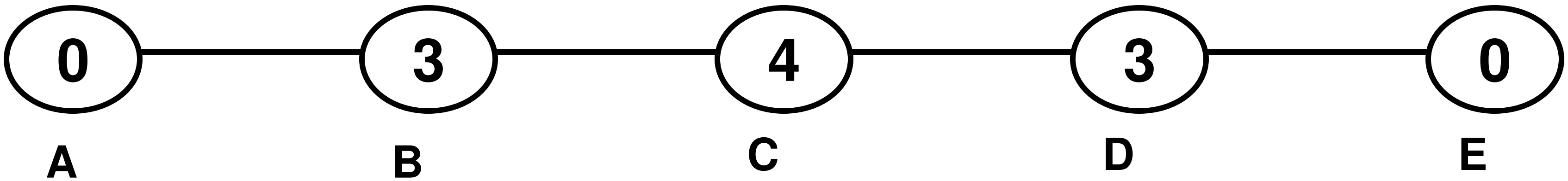
$$C_c(i) = \left[\sum_{j=1}^{N} d(i,j) \right]^{-1}$$

d(i, j) is the hop count on the shortest path from node i to node j.



The shaded node has a high closeness as it is just one hop away from the most central node and can reach the rest of the network in fewer hops.

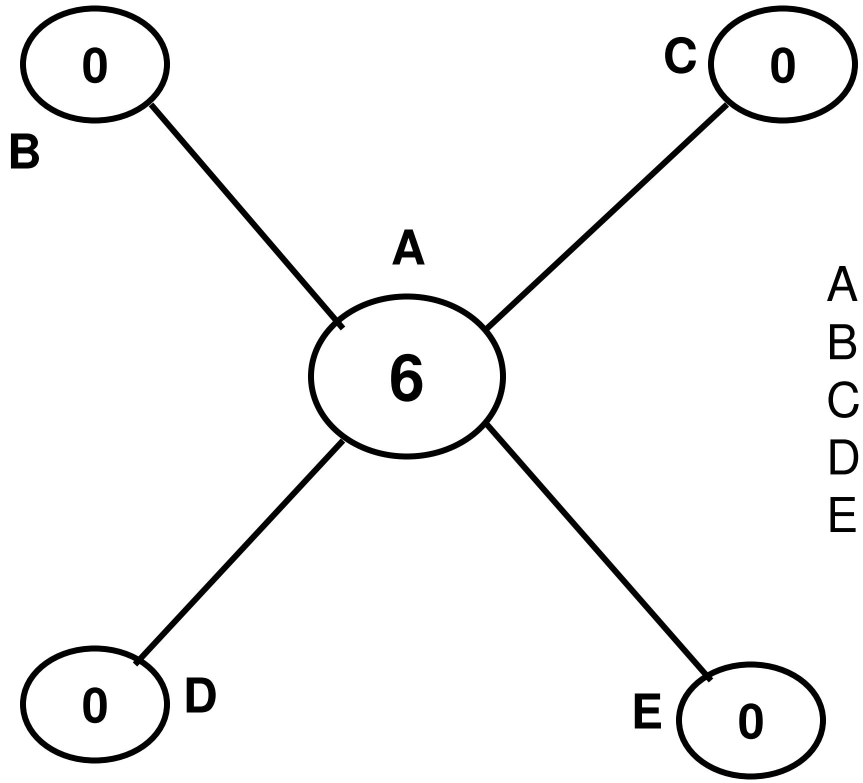
Closeness: Toy Example (1)



	A	B	C	D	E	Sum	Closeness
A	0	1	2	3	4	10	0.1
B	1	0	1	2	3	7	0.14
C	2	1	0	1	2	6	0.17
D	3	2	1	0	1	7	0.14
E	4	3	2	1	0	10	0.1

$$\text{Closeness Centrality} = \frac{[\,(0.17-0.17)*1 + (0.17-0.14)*2 + (0.17-0.1)*2\,]}{[\,(5-1)(5-2)\,]} = 0.017$$

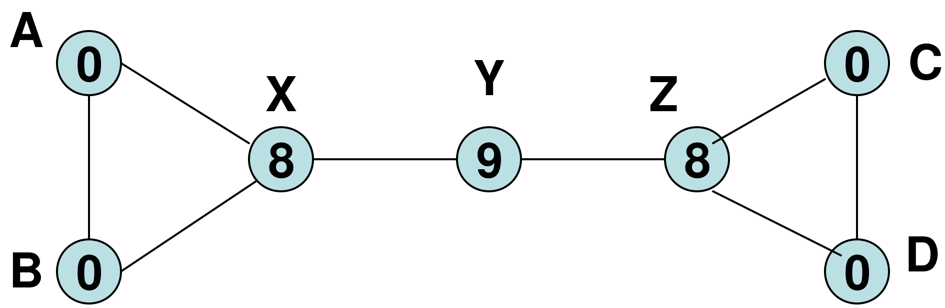
Closeness on Toy Networks (2)



	A	B	C	D	E	Sum	Closeness
A	0	1	1	1	1	4	0.25
B	1	0	2	2	2	7	0.14
C	1	2	0	2	2	7	0.14
D	1	2	2	0	2	7	0.14
E	1	2	2	2	0	7	0.14

$$\text{Closeness Centralization} = \frac{[\,(0.25 - 0.25)*1 + (0.25 - 0.17)*4\,]}{[\,(5-1) * (5-2)\,]} = 0.0267$$

Closeness on Toy Networks (3)



	A	B	C	D	X	Y	Z	Sum	Closeness
A	0	1	4	4	1	2	3	15	0.067
B	1	0	4	4	1	2	3	15	0.067
C	4	4	0	1	3	2	1	15	0.067
D	4	4	1	0	3	2	1	15	0.067
X	1	1	3	3	0	1	2	11	0.091
Y	2	2	2	2	1	0	1	10	0.1
Z	3	3	1	1	2	1	0	11	0.091

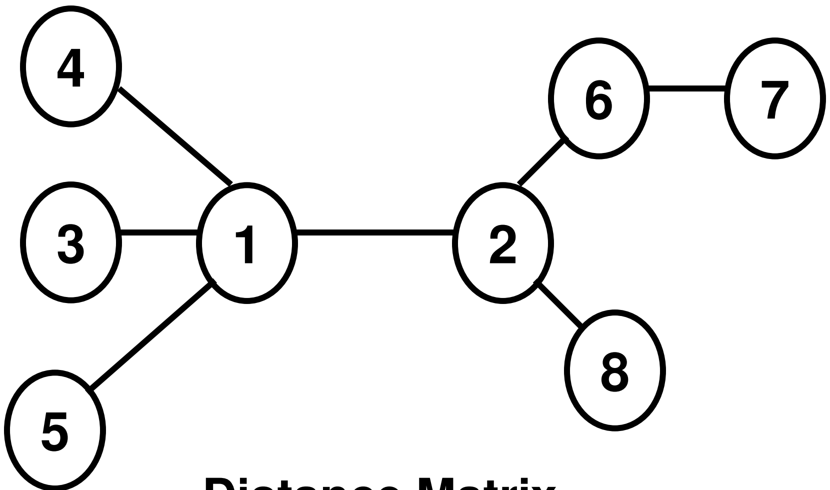
$$\text{Closeness Centralization} = \frac{[\,(0.1 - 0.1)*1 + (0.1 - 0.067)*4 + (0.1 - 0.091)*2\,]}{[\,(7-1) * (7-2)\,]}$$

$= 0.005$

Farness Centrality

- The closeness centrality equally ranks nodes that have the same sum of the distances to every other node.
  - However, the closeness centrality does not take into consideration the variations in the distances of a node to every other node.
- Farness Centrality prefers nodes with the smallest total distance to every other node and if there is a tie, it gives preference to nodes that have less variations in the individual distances of the vying node to every other node.
- Farness Centrality of a node i is the value in the ith entry of the Principal Eigen Vector (the Eigen Vector corresponding to the largest Eigen Value) of the Distance Matrix that captures the # hops between any two nodes in the network.

Farness Centrality: Example



Distance Matrix

	1	2	3	4	5	6	7	8
1	**0**	**1**	**1**	**1**	**1**	**2**	**3**	**2**
2	**1**	**0**	**2**	**2**	**2**	**1**	**2**	**1**
3	**1**	**2**	**0**	**2**	**2**	**3**	**4**	**3**
4	**1**	**2**	**2**	**0**	**2**	**3**	**4**	**3**
5	**1**	**2**	**2**	**2**	**0**	**3**	**4**	**3**
6	**2**	**1**	**3**	**3**	**3**	**0**	**1**	**2**
7	**3**	**2**	**4**	**4**	**4**	**1**	**0**	**3**
8	**2**	**1**	**3**	**3**	**3**	**2**	**3**	**0**

$\eta 1 = 16.315$

$\delta 1 =$
[0.6717
0.6691
1.0023
1.0023
1.0023
0.8711
1.1799
1]

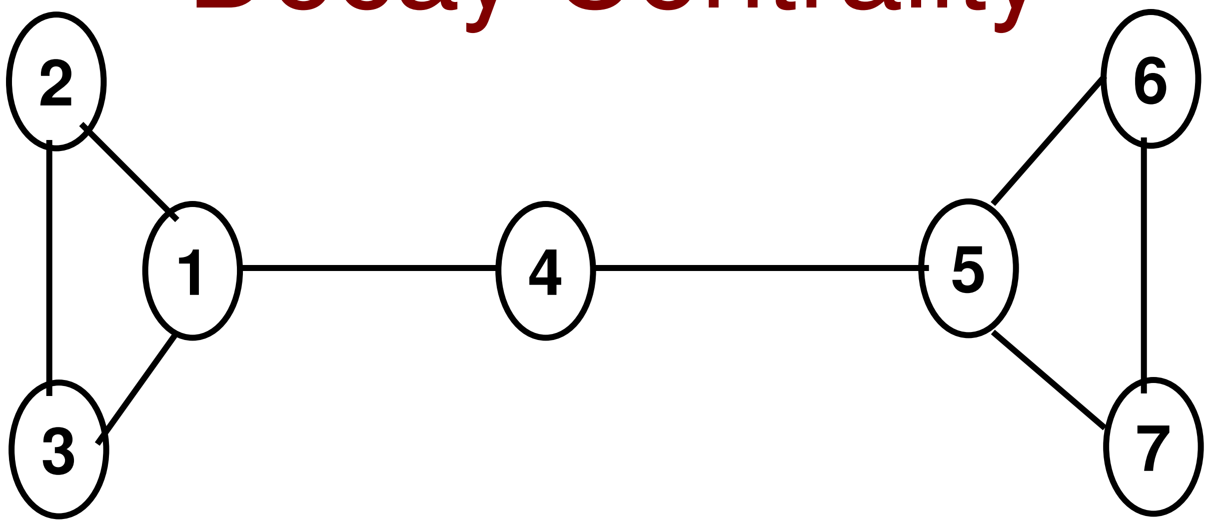
Ranking of Nodes

Score	Node ID
0.6691	2
0.6717	1
0.8711	6
1	8
1.0023	3
1.0023	4
1.0023	5
1.1799	7

Decay Centrality

- Incorporates the value of the connection to the other nodes.
- With a parameter δ, the value decays with distance as δ^1, δ^2, δ^3, …, where 1, 2, 3, …, are the distances of the node to the other nodes.
- $C_i^{dec}(G) = \Sigma_{j \neq i} \delta^{dist(i, j)}$
  - If δ is close to 1, decay centrality becomes a measure of the component size (each node contributes a value closer to 1)
  - If δ is close to 0, decay centrality becomes a measure of the node degree (the contributions of the higher order terms of δ become negligible)
  - Intermediate values of δ capture the decay centrality (weigh the indirect connections less than the direct connections).
- Captures the importance of a node being closer to many nodes.

Decay Centrality



	Node 4	Node 1	Node 2
δ = 0.25	0.75	0.8438	0.5859
δ = 0.50	2.0	2.0	1.50
δ = 0.75	3.75	3.656	3.117
δ = 0.85	4.59	4.5	4.08
δ = 0.95	5.51	5.46	5.29

For the perceived node with high centrality (node 4), the intermediate values of δ = 0.5, 0.75 result in more nodes contributing to a lower order exponent for δ, leading to a smaller decay, and an overall high Centrality score.

Assortativity

- Assortativity or Assortative mixing is a preference for a network's nodes to attach to others that are similar in some way.
  - Often examined with respect to node degree
  - In social networks, highly connected nodes tend to be connected with other high degree nodes
- Disassortativty: High-degree nodes attach to low-degree nodes and vice-versa.
  - Technological and biological networks typically show disassortative mixing

- Assortativity is measured in the form of Correlation Coefficient between the cause and effect.
- Assortativity Coefficient Computation for Node Degree

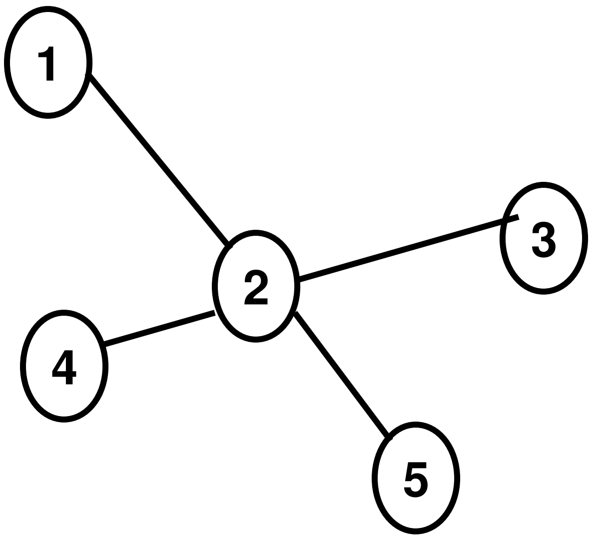
$$r_{XY} = \frac{\sum_{i=1}^{m}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{m}(X_i - \overline{X})^2}\sqrt{\sum_{i=1}^{m}(Y_i - \overline{Y})^2}}$$

m – the number of edges

(Xi, Yi) – the degrees of endpoints of edge i

$(\overline{X}, \overline{Y})$: the mean of the degrees of the endpoints of edge i

Assortativity Example (1)



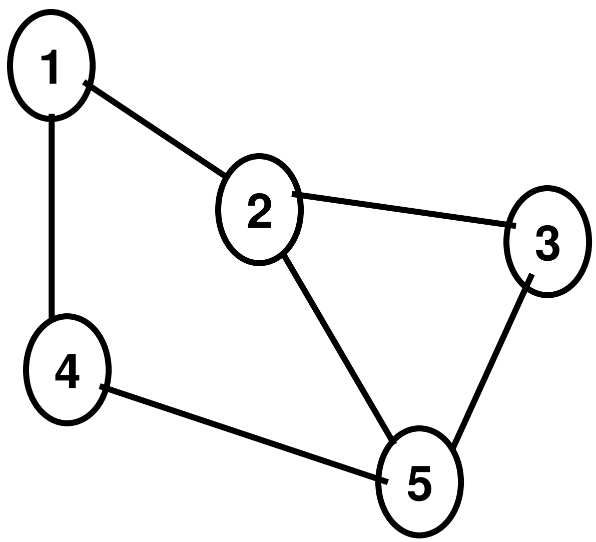
	1	2	3	4	5
1	0	1	0	0	0
2	1	0	1	1	1
3	0	1	0	0	0
4	0	1	0	0	0
5	0	1	0	0	0

Node ID	Degree
1	1
2	4
3	1
4	1
5	1

Edge	Xi	Yi	Xi-AvgX	Yi-AvgY	(Xi-AvgX)(Yi-AvgY)
1-2	1	4	-2.25	2.25	-5.063
2-3	4	1	0.75	-0.75	-0.563
2-4	4	1	0.75	-0.75	-0.563
2-5	4	1	0.75	-0.75	-0.563
Avg X	3.25	1.75			
Sum (sample-mean)^2			6.75	6.75	

Pearson Correlation Coefficient = -1.0 **Disassortative**

Assortativity Example (2)



	1	2	3	4	5
1	0	1	0	1	0
2	1	0	1	0	1
3	0	1	0	0	1
4	1	0	0	0	1
5	0	1	1	1	0

Node ID	Degree
1	2
2	3
3	2
4	2
5	3

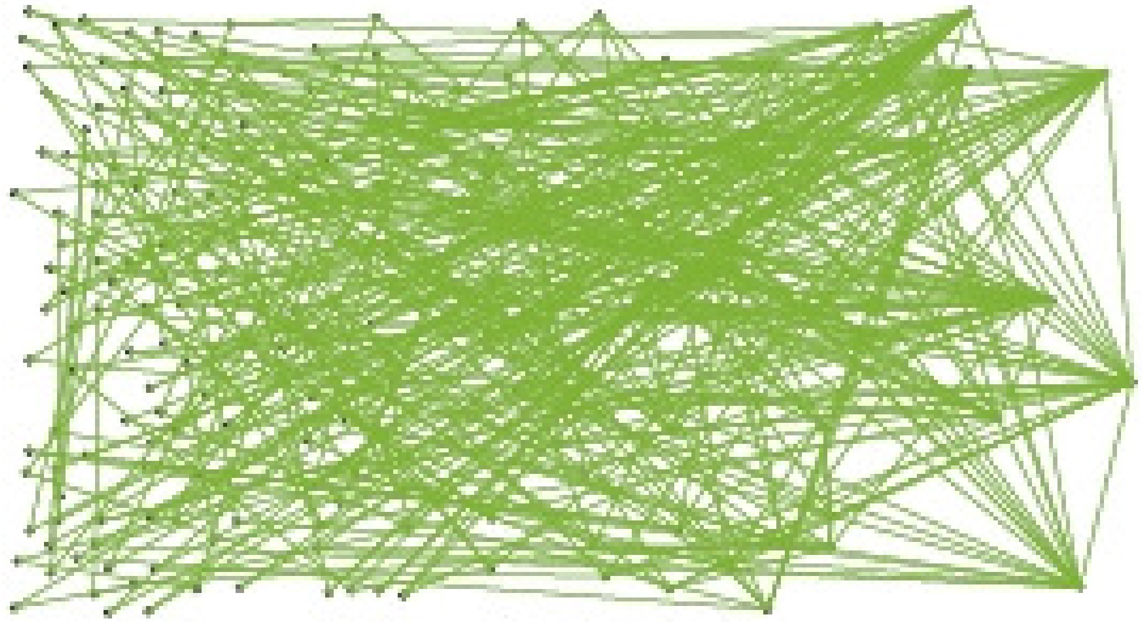
Edge	Xi	Yi	Xi-AvgX	Yi-AvgY	(Xi-AvgX)(Yi-AvgY)
1-2	2	3	-0.33	0.33	-0.1089
1-4	2	2	-0.33	-0.67	0.2211
2-3	3	2	0.67	-0.67	-0.4489
2-5	3	3	0.67	0.33	0.2211
3-5	2	3	-0.33	0.33	-0.1089
4-5	2	3	-0.33	0.33	-0.1089
Avg X	2.33	2.67			
Sum (sample-mean)^2			1.3334	1.3334	

Pearson Correlation Coefficient = -0.25 **Non-assortative**

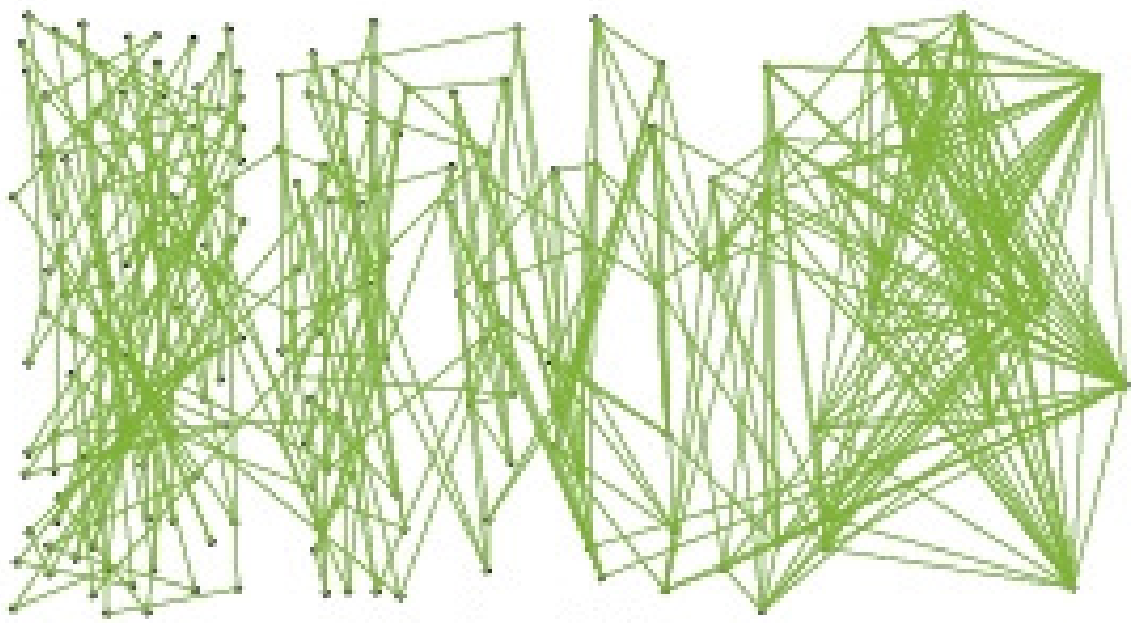
Assortativity of Real-World Networks

Network	n	r
Physics coauthorship (a)	52 909	0.363
Biology coauthorship (a)	1 520 251	0.127
Mathematics coauthorship (b)	253 339	0.120
Film actor collaborations (c)	449 913	0.208
Company directors (d)	7 673	0.276
Internet (e)	10 697	−0.189
World-Wide Web (f)	269 504	−0.065
Protein interactions (g)	2 115	−0.156
Neural network (h)	307	−0.163
Marine food web (i)	134	−0.247
Freshwater food web (j)	92	−0.276
Random graph (u)		0
Barabási and Albert (w)		0

Assortativity of Scale-Free Networks



$\mathcal{A} = 0$



$\mathcal{A} = 0.43$

Link Analysis-based Ranking

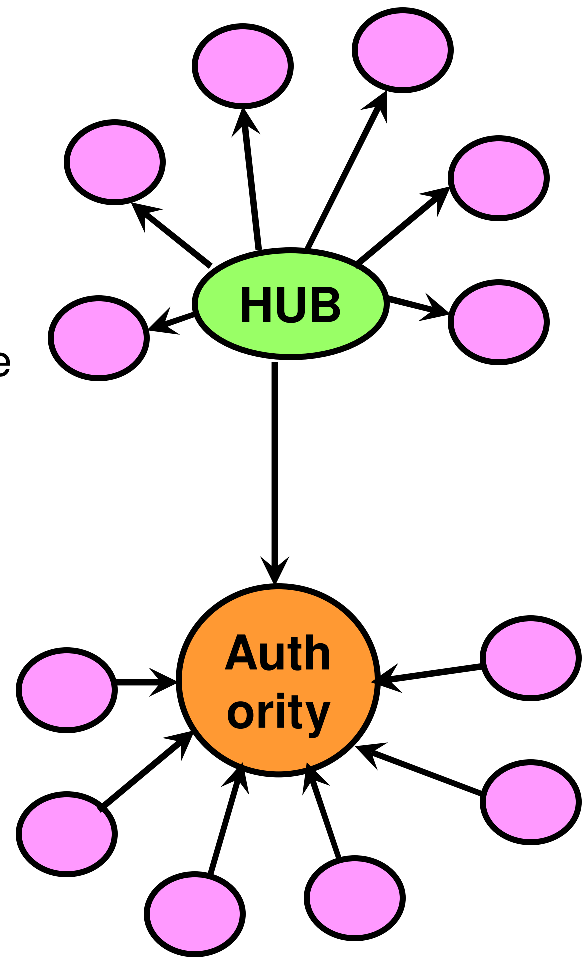
- We want to rank a node in a graph based on the number of edges pointing to it and/or leaving it as well as based on the rank of the nodes at the other end of these edges.

- Used primarily in web search
  – We model the web as a graph: the pages as nodes and the edges are directed edges – a page citing (having a link to) another page.

- Hubs and Authorities (HITS) algorithm
- PageRank algorithm

Hypertext Induced Topic Search (HITS) Algorithm

- **Hub:** Node that points to lots of pages
  - Yahoo like directory
- **Authority:** Node to which several other nodes point to
  - The larger the number of nodes pointing to a node, the more authoritative is the view presented by a node on a particular subject
- The HITS algorithm assigns **two scores for each page**:
  - **Authority:** an estimate of the value of the contents of the page
  - **Hub:** an estimate of the value of its links to other pages



- A page is considered to be **more authoritative** if it is referenced by many hub pages that are relevant to a search query
- A page is a **hub page** for a search query if it points to many authoritative pages for that query

- Good **authoritative** and **hub** pages reinforce one another.

A variant of HITS is used by Ask.com

Finding Pages for a Query in HITS

- **Initial Work**
- Step 1: Submit query *q* to a similarity-based engine and record the top *n*, i.e., the root set RS(q) pages.
- Step 2: Expand set RS(q) into the base set BS(q) to include pages pointed by RS(q) pages
- Step 3: Also include into BS(q), the pages pointing to RS(q) pages.
- **Run the HITS algorithm**
  - For each page pj, compute the authority and hub score of pj through a sequence of iterations.
- **After obtaining the final authority and hub scores** for each page, display the search results in the decreasing order of the authority scores. Pages having zero authority scores (nodes with no incoming links – strictly hubs) are listed in the decreasing order of their hub scores.
  - Note: nodes that are strictly hubs still contribute to the authority of the nodes that it points to.

HITS Algorithm

- Let E be the set of links in BS(q) and a link from page pi to pj is denoted by the pair (i, j).
- A: Authority Update Step

$$a(p_j) = \sum_{(i,j)\in E} h(p_i)$$

H: Hub Update Step

$$h(p_j) = \sum_{(j,k)\in E} a(p_k)$$

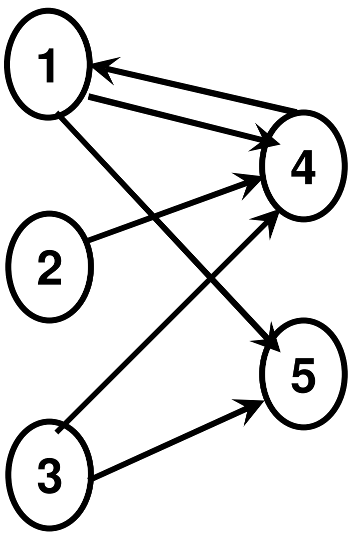
- After each iteration i, we scale the 'a' and 'h' values:

$$a^{(i)}(p_j) = \frac{a^{(i)}(p_j)}{\sqrt{\sum_k \left(a^{(i)}(p_k)\right)^2}}$$

$$h^{(i)}(p_j) = \frac{h^{(i)}(p_j)}{\sqrt{\sum_k \left(h^{(i)}(p_k)\right)^2}}$$

- As can be noted above, the two steps are interwined: one uses the values computed from the other.
  - In this course, we will follow the asynchronous mode of computation, according to which the authority values are updated first for a given iteration i and then the hub values are updated.
    - The hub values at iteration *i* are using the authority values just computed in iteration *i* (rather than iteration *i* – 1).

HITS Example (1)



Initial
a = [1 1 1 1 1] h = [1 1 1 1 1]

It # 1
a = [1 0 0 3 2] h = [5 3 5 1 0]
After Normalization,
a = [0.26 0 0 0.80 0.53] h = [0.64 0.38 0.64 0.12 0]

It # 2
a = [0.12 0 0 1.66 1.28] h = [2.94 1.66 2.94 0.12 0]
After Normalization,
a = [0.057 0 0 0.79 0.61] h = [0.66 0.37 0.66 0.027 0]

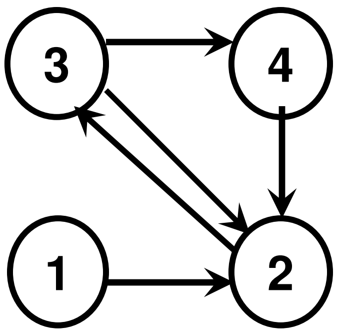
It # 3
a = [0.027 0 0 1.69 1.32] h = [3.01 1.69 3.01 0.027 0]
After Normalization,
a = [0.0126 0 0 0.79 0.61] h = [0.66 0.37 0.66 0.006 0]

It # 4
a = [0.006 0 0 1.69 1.32] h = [3.01 1.69 3.01 0.006 0]
After Normalization,
a = [0.003 0 0 0.79 0.61] h = [0.66 0.37 0.66 0.001 0]

Order Pages Listed after Search

4
5
1
3
2

HITS Example (2)



Initial
a = [1 1 1 1] h = [1 1 1 1]

It # 1
a = [0 3 1 1] h = [3 1 4 3]
After Normalization,
a = [0 0.91 0.30 0.30] h = [0.51 0.17 0.68 0.51]

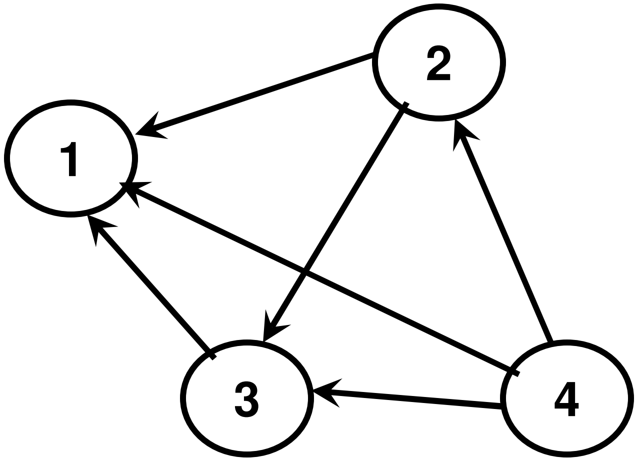
It # 2
a = [0 1.70 0.17 0.68] h = [1.70 0.17 2.38 1.70]
After Normalization,
a = [0 0.92 0.09 0.37] h = [0.50 0.05 0.70 0.50]

It # 3
a = [0 1.70 0.05 0.70] h = [1.70 0.05 2.4 1.70]
After Normalization,
a = [0 0.92 0.027 0.38] h = [0.50 0.014 0.70 0.50]

It # 4
a = [0 1.70 0.014 0.70] h = [1.70 0.014 2.4 1.70]
After Normalization,
a = [0 0.92 0.008 0.38] h = [0.50 0.004 0.71 0.50]

Order Pages Listed after Search

2
4
3
1



HITS Example (3)

Initial
a = [1 1 1 1] **h = [1 1 1 1]**

It # 1
a = [3 1 2 0] **h = [0 5 3 6]**
After Normalization,
a = [0.80 0.27 0.53 0] **h = [0 0.59 0.36 0.72]**

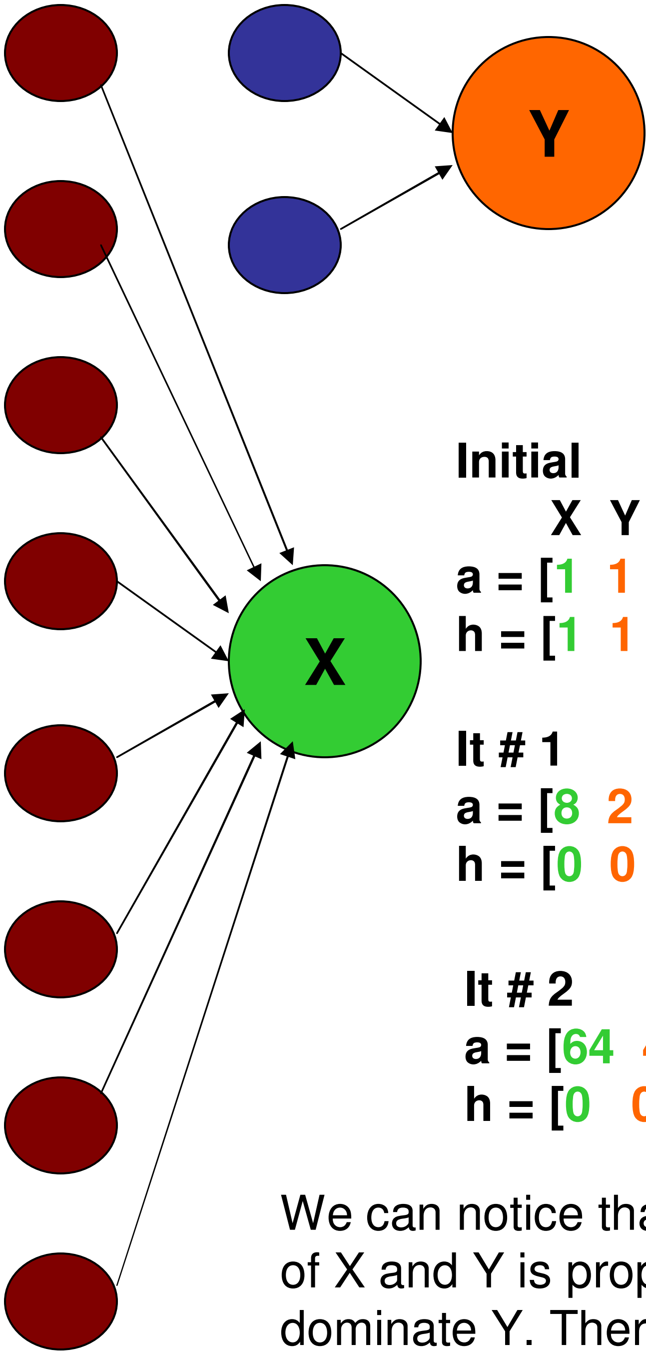
Order Pages Listed after Search

1
3
2
4

It # 2
a = [1.67 0.72 1.31 0] **h = [0 2.98 1.67 3.7]**
After Normalization,
a = [0.745 0.32 0.58 0] **h = [0 0.59 0.33 0.73]**

It #3
a = [1.65 0.73 1.32 0] **h = [0 2.97 1.65 3.7]**
After Normalization,
a = [0.74 0.32 0.59 0] **h = [0 0.59 0.33 0.73]**

HITS Example (4)



- Assume 'x' web-pages point to page X and 'y' pages point to page Y, where x >> y. What happens with the hubs and authority values of X and Y respectively?
- Assume no normalization is done at the end of each iteration.

Initial

X Y ←x web-pages →<-y ->

a = [1 1 1 1 1 1 1 1 1 1 1 1]

h = [1 1 1 1 1 1 1 1 1 1 1 1]

It # 1

a = [8 2 0 0 0 0 0 0 0 0 0 0]

h = [0 0 8 8 8 8 8 8 8 8 2 2]

It # 2

a = [64 4 0 0 0 0 0 0 0 0 0 0]

h = [0 0 64 64 64 64… 64 4 4]

We can notice that with each iteration i, the ratio of the authority values of X and Y is proportional to (x/y)^i. After a while, X will completely dominate Y. There is no change in the hub values of X and Y though.

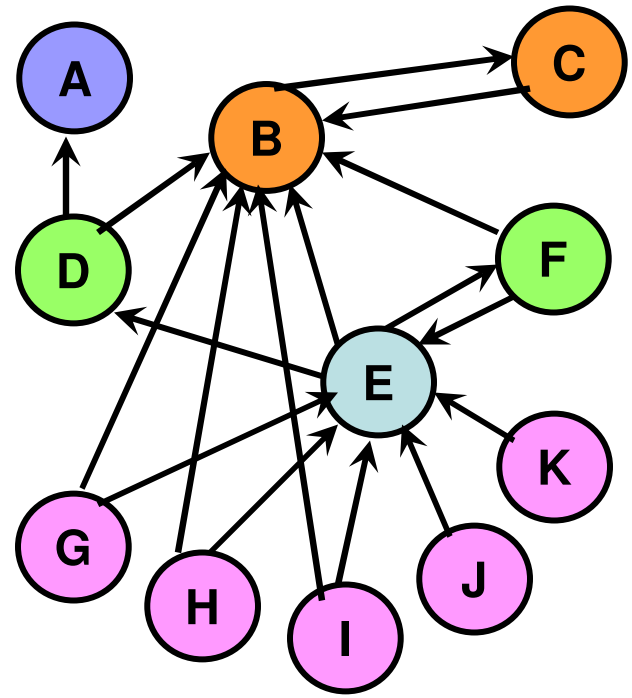
PageRank

- The basic idea is to analyze the link structure of the web to figure out which pages are more authoritative (important) in terms of quality.
- It is a content-independent scheme.
- If Page A has a hyperlink to Page B, it can be considered as a vote of A for B.
  - If multiple pages link to B, then page B is likely to be a good page.
- A page is likely to be good if several other good pages link to it (a bit of recursive definition).
  - Not all pages that link to B are of equal importance.
  - A single link from CNN or Yahoo may be worth several times
- The web pages are first searched based on the content. The retrieved web pages are then listed based on their rank (computed on the original web, unlike HITS that is run on a graph of the retrieved pages).
- The Page Rank of the web pages are indexed (recomputed) for every regular time period.

PageRank (Random Web Surfer)

- Web – graph of pages with the hyperlinks as directed edges.
- Analogy used to explain PageRank algorithm (Random Web Surfer)
- User starts browsing on a random page
- Picks a random out-going link listed in that page and goes there (with a probability 'd', also called damping factor)
  - Repeated forever
- The surfer jumps to a random page with probability 1-d.
  - Without this characteristic, there could be a possibility that someone could just end up oscillating between two pages B and C as in the traversing sequence below for the graph shown aside:

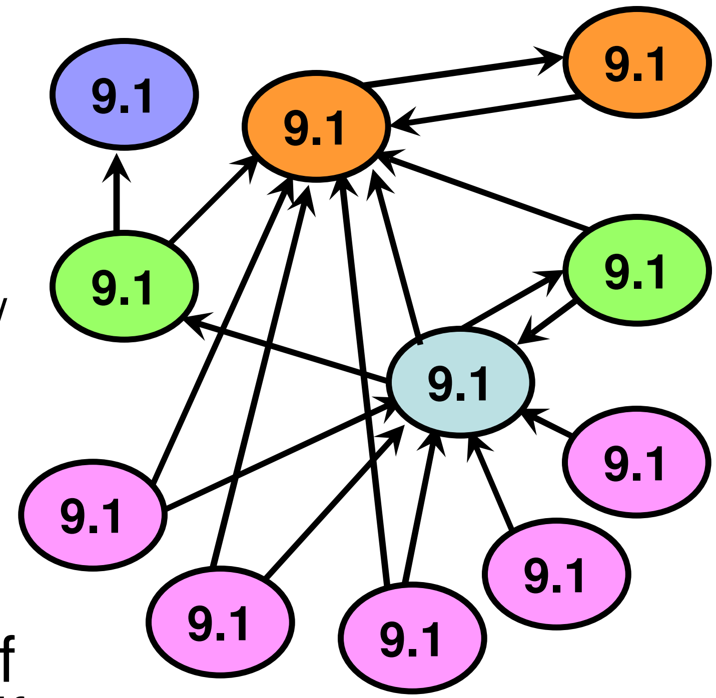
    G → E → F → E → D → B → C



Lets say d = 0.85.
To decide the next page to move, the surfer simply generates a random number, r. If r <= 0.85, then the surfer randomly chooses an out-going link from the existing page. Otherwise, jumps to a randomly chosen page among all the pages, including the current page.

PageRank Algorithm

- PageRank of Page X is the probability that the surfer is at page X at a randomly selected time.
  - Basically the proportion of time, the surfer would spend at page X.
- **PageRank Algorithm**
- **Initial:** Every node in the graph gets the same pagerank. PR(X) = 100% / N, where N is the number of nodes.
- At any time, at the end of each iteration, the page rank of all nodes add up to 100%.
- Actually, the initial pagerank value of a node is the pagerank at any time, if there are no edges in the graph. We have 100% / N chance of jumping to any node in the graph at any time.



Initial PageRank of Nodes

PageRank Algorithm

Page Rank of Node X

$$PR(x) = \frac{(1-d)*100}{N} + d\sum_{y->x}\frac{PR(y)}{Out(y)}$$

Assuming there are NO Sink nodes

- Page Rank of Node X is the probability of being at node X at the current time.
- How can we visit node X from where we are?
  - **(1-d) term: Random Jump:** The probability of ending up at node X because of a random jump from some node, including node X, is 1/N.
  - However, such a random jump itself could occur with a probability of (1-d).
  - This amounts to a probability of (1-d)/N to be at node X due to a random jump.

PageRank Algorithm

Page Rank of Node X

$$PR(x) = \frac{(1-d)*100}{N} + d\sum_{y->x}\frac{PR(y)}{Out(y)}$$

Assuming there are NO Sink nodes

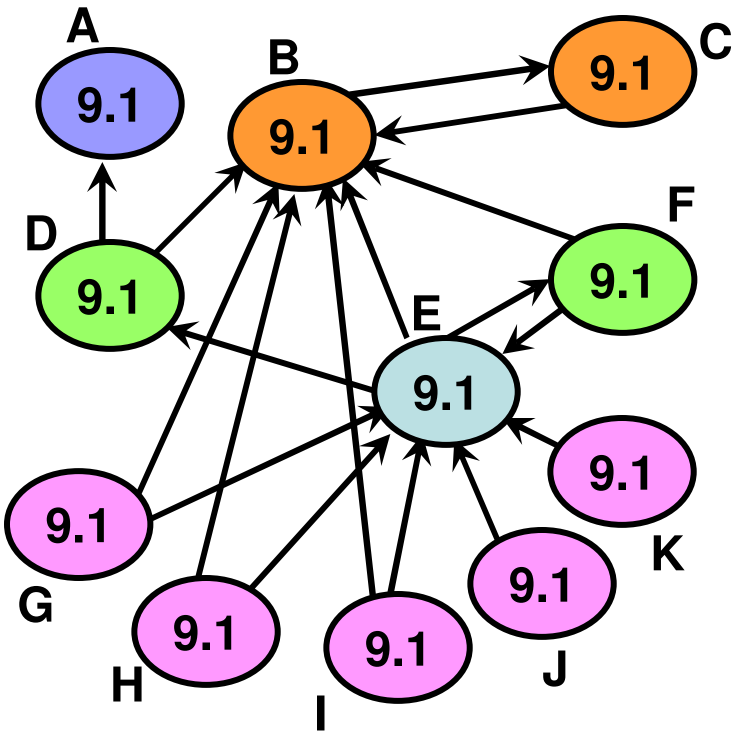
- Page Rank of Node X is the probability of being at node X at the current time.
- How can we visit node X from where we are?
  - **d term: Edge Traversal from a Neighbor:**
  - We could visit node X from one of the nodes that point to node X.
  - Lets say, we are at node Y in the previous iteration. The probability of being at node Y in the previous iteration is PR(Y). We can visit any of Y's neighbors.
  - The probability of visiting node X among the Out(Y) out-going links of node Y is PR(Y) * (1 / Out(Y)) = PR(Y) / Out(Y).
  - Likewise, we could visit X from any of its neighbors.
  - All the probabilities of visiting X from any of its neighbors have to be added, because visiting X from any of its neighbors is independent of the neighbors.
  - The whole event of visiting from a neighbor occurs with a prob. 'd'

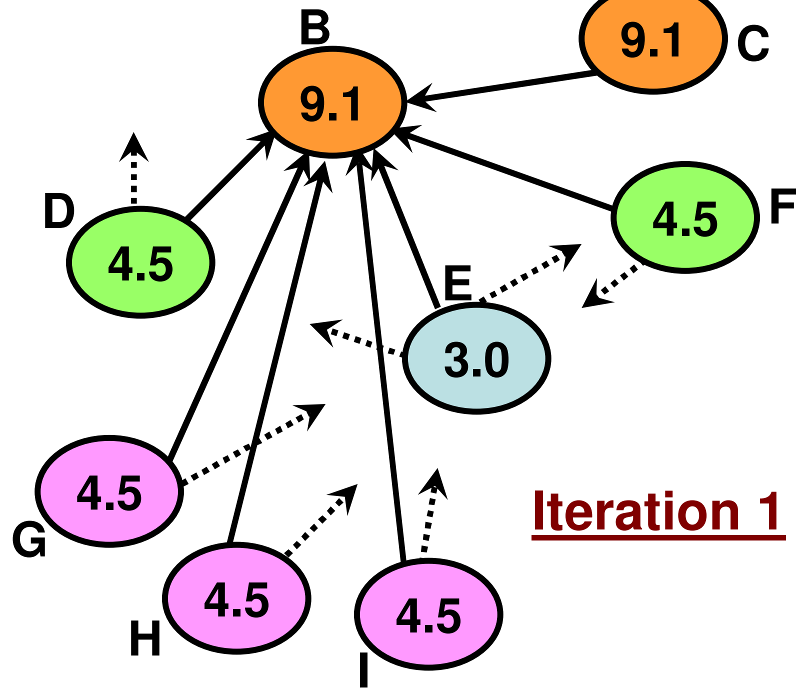
PageRank

- Since Page Rank PR(X) denotes the probability of being at node X at any time, the sum of the Page Ranks of all the nodes at any time should be equal to 1.
- We can also interpret the traversal from a node Y to node X as node Y contributing a part of its PR to node X (node Y equally shares its PR to the nodes connected to it through its out-going links).
- Implementation:
  – Note that (unlike HITS) we need to use the page rank values of the nodes from the previous iteration to update the page rank values of the nodes in the current iteration.
    - **Need to maintain two arrays at any time t: $PR^{(t-1)}$ and $PR^{(t)}$**

Calculating PageRank of Node B

Initial PageRank of Nodes





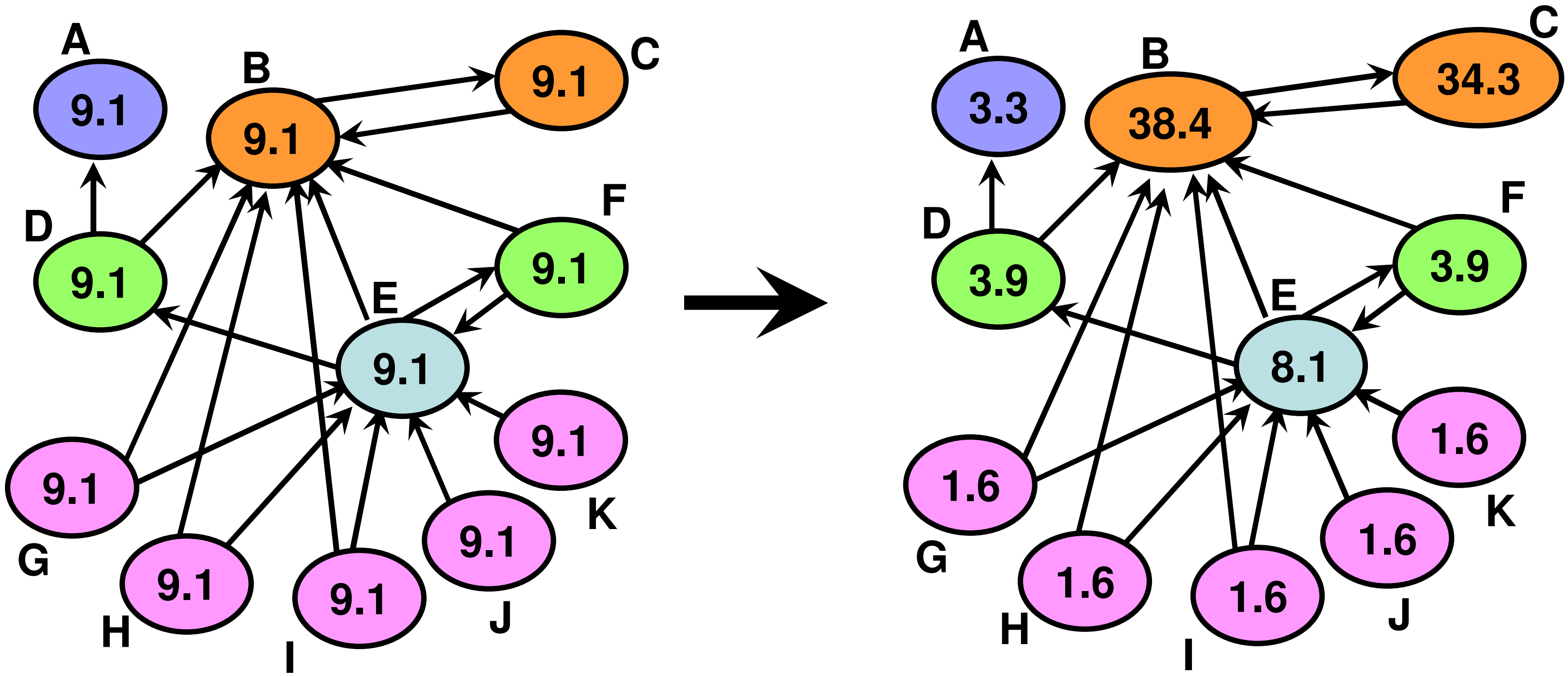
Assume the damping factor d = 0.85

For any iteration,

PR(B) = 0.15 * 9.1 +
0.85 * [PR(C) + ½ PR(D) +
⅓ PR(E) + ½ PR(F) +
½ PR (G) + ½ PR(H) + ½ PR(I)]

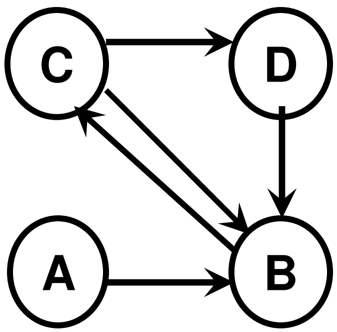
For Iteration 1,
Substituting the PR values of the nodes (initial values),
we get PR(B) ≈ 31

Final PageRank Values for the Sample Graph



PageRank: More Observations

- Algorithm converges (few iterations sufficient)
- For an arbitrary graph, it is pretty difficult to figure out the final page rank values of the nodes.
- Certain inferences could be however made.
- For our sample graph:
  - For nodes that do not have any in-links pointing to them, the only way we will end up at these nodes is through a random jump: this happens with a probability (1-d)/N. In our case, it is (1-0.85)* 100/11 = 1.6%.
  - Two nodes with links from the same node (symmetric in-links) have the same PR. (nodes D and F) and it will be higher than those nodes without any in-links.
  - One in-link from a node with high PR value contributes significantly to the PR value of a node compared to the in-links from several low PR nodes.
    - In our sample graph, an in-link from node B contributes significantly for node C compared to the several in-links that node E gets from the low-PR nodes. So, the quality of the in-links matters more than the number of in-links.



Note that there are NO sink nodes (nodes without any out-going links)

Assume damping Factor d = 0.85

PR(A) = (1-d)*100/4
PR(B) = (1-d)*100/4 + d*[PR(A) + 1/2 * PR(C) + PR(D)]
PR(C) = (1-d)*100/4 + d*[PR(B)]
PR(D) = (1-d)*100/4 + d*[1/2*PR(C)]

Initial
PR(A) = 25
PR(B) = 25
PR(C) = 25
PR(D) = 25

It # 1
PR(A) = 3.75
PR(B) = 56.88
PR(C) = 25
PR(D) = 14.38

It # 2
PR(A) = 3.75
PR(B) = 29.79
PR(C) = 52.10
PR(D) = 14.38

It # 3
PR(A) = 3.75
PR(B) = 41.30
PR(C) = 29.07
PR(D) = 25.89

It # 4
PR(A) = 3.75
PR(B) = 41.29
PR(C) = 38.86
PR(D) = 16.10

It # 5
PR(A) = 3.75
PR(B) = 37.14
PR(C) = 38.85
PR(D) = 20.27

It # 6
PR(A) = 3.75
PR(B) = 40.68
PR(C) = 35.32
PR(D) = 20.26

It # 7
PR(A) = 3.75
PR(B) = 39.17
PR(C) = 38.33
PR(D) = 18.76

It # 8
PR(A) = 3.75
PR(B) = 39.17
PR(C) = 37.04
PR(D) = 20.04

It # 9
PR(A) = 3.75
PR(B) = 39.71
PR(C) = 37.04
PR(D) = 19.49

It # 10
PR(A) = 3.75
PR(B) = 39.25
PR(C) = 37.5
PR(D) = 19.49

Ranking
B
C
D
A

Page Rank Example (1)

Page Rank: Graph with Sink Nodes Motivating Example

- Consider the graph: A → B
- Let d = 0.85
- PR(A) = 0.15*100/2 PR(B) = 0.15*100/2 + 0.85*PR(A)
- Initial: PR(A) = 50, PR(B) = 50
- Iteration 1:
  - PR(A) = 0.15*100/2 = 7.5
  - PR(B) = 0.15*100/2 + 0.85 * 50 = 50.0
  - PR(A) + PR(B) = 57.5
  - Note that the PR values do not add up to 1.
  - This is because, B is not giving back the PR that it receives from A to any other node in the graph. The (0.85*50 = 42.5) value of PR that B receives from A is basically lost.
  - Once we get to B, there is no way to get out of B other than random jump to A and this happens only with probability (1-d).

Page Rank: Sink Nodes (Solution)

- Assume implicitly that the sink node is connected to every node in the graph (including itself).
  - The sink node equally shares its PR with every node in the graph, including itself.
  - If *z* is a sink node, with the above scheme, out(z) = N, the number of nodes in the graph.
- The probability of getting to node X at a given time is still the two terms below:
  - Random jump from any node (probability, 1-d)
  - Visit from a node with in-link to node X (probability, d)

Page Rank of Node X

$$PR(x) = \frac{(1-d)*100}{N} + d\sum_{y->x}\frac{PR(y)}{Out(y)} + \frac{d}{N}\sum_{z->\varphi}PR(z)$$

Explicit out-going links to certain nodes

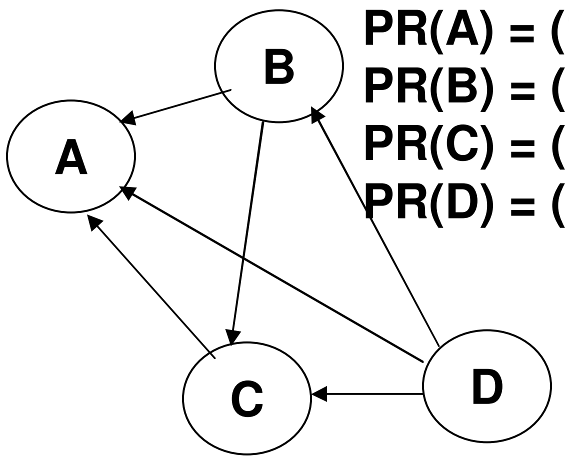
Implicit out-going links to all nodes (sink nodes)

the second term of the original Page Rank formula is now broken between that of nodes with explicit out-going links to one or more selected nodes and the sink nodes with implicit out-going links to all nodes.

Consolidated PageRank Formula

$$PR(x) = \frac{(1-d)*100}{N} + d\sum_{y->x}\frac{PR(y)}{Out(y)} + \frac{d}{N}\sum_{z->\varphi} PR(z)$$

Page Rank Example (2)

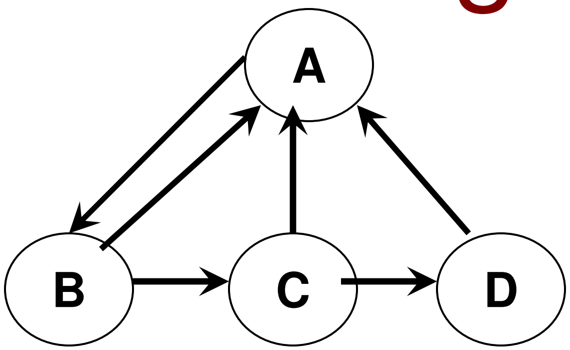


PR(A) = (1-d)*100/4 + d [PR(B)/2 + PR(C)/1 + PR(D)/3] + (d/4)*[PR(A)]
PR(B) = (1-d)*100/4 + d [PR(D)/3] + (d/4)*[PR(A)]
PR(C) = (1-d)*100/4 + d [PR(B)/2 + PR(D)/3] + (d/4)*[PR(A)]
PR(D) = (1-d)*100/4 + (d/4)*[PR(A)]

Node Ranking: A, C, B, D

	Initial	It # 1	It # 2	It # 3	It # 4
PR(A)	25	48.02	46.14	44.41	45.32
PR(B)	25	16.15	16.52	17.51	17.03
PR(C)	25	26.77	23.386	24.53	24.47
PR(D)	25	9.063	13.954	13.55	13.18

Page Rank Example (3)



PR(A) = (1-d)*100/4 + d*[½*PR(B) + ½*PR(C) + PR(D)]
PR(B) = (1-d)*100/4 + d*[PR(A)]
PR(C) = (1-d)*100/4 + d*[½*PR(B)]
PR(D) = (1-d)*100/4 + d*[½*PR(C)]

	Initial	It # 1	It # 2	It # 3	It # 4
PR(A)	25	46.25	32.71	36.54	34.91
PR(B)	25	25	43.06	31.55	34.81
PR(C)	25	14.38	14.38	22.05	17.16
PR(D)	25	14.38	9.86	9.86	13.12

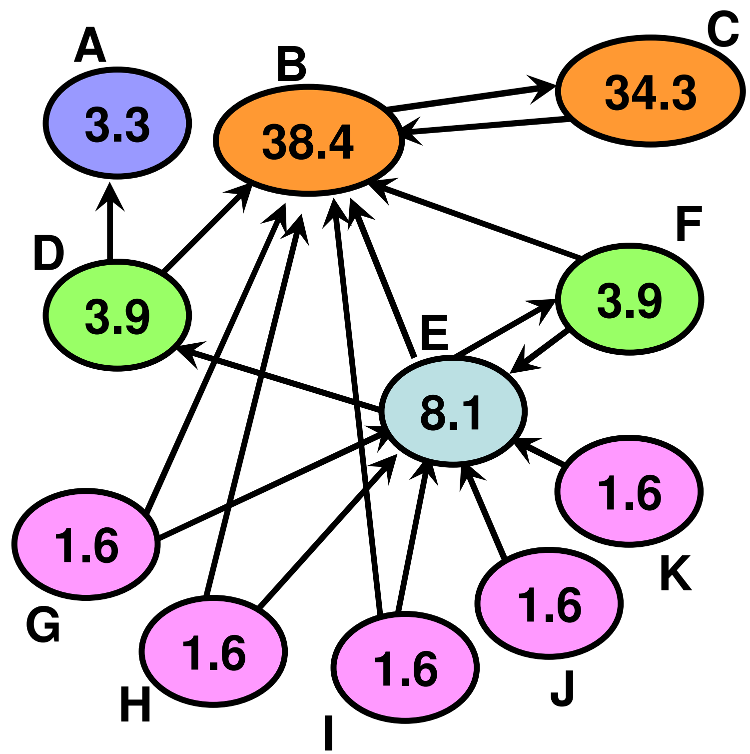
	It # 5	It # 6	It # 7	It # 8	It # 9
PR(A)	36.99	35.22	36.19	35.68	36.03
PR(B)	33.42	35.12	33.68	34.51	34.08
PR(C)	18.54	17.95	18.68	18.06	18.42
PR(D)	11.04	11.63	11.38	11.69	11.43

Node Ranking: A B C D

Computing Huffman Codes for Nodes using their PageRank Values

A	3.3
B	38.4
C	34.3
D	3.9
E	8.1
F	3.9
G	1.6
H	1.6
I	1.6
J	1.6
K	1.6

B	0
C	11
K	10000
I	100010
J	100011
A	10011
G	100100
H	100101
D	10100
F	10101
E	1011



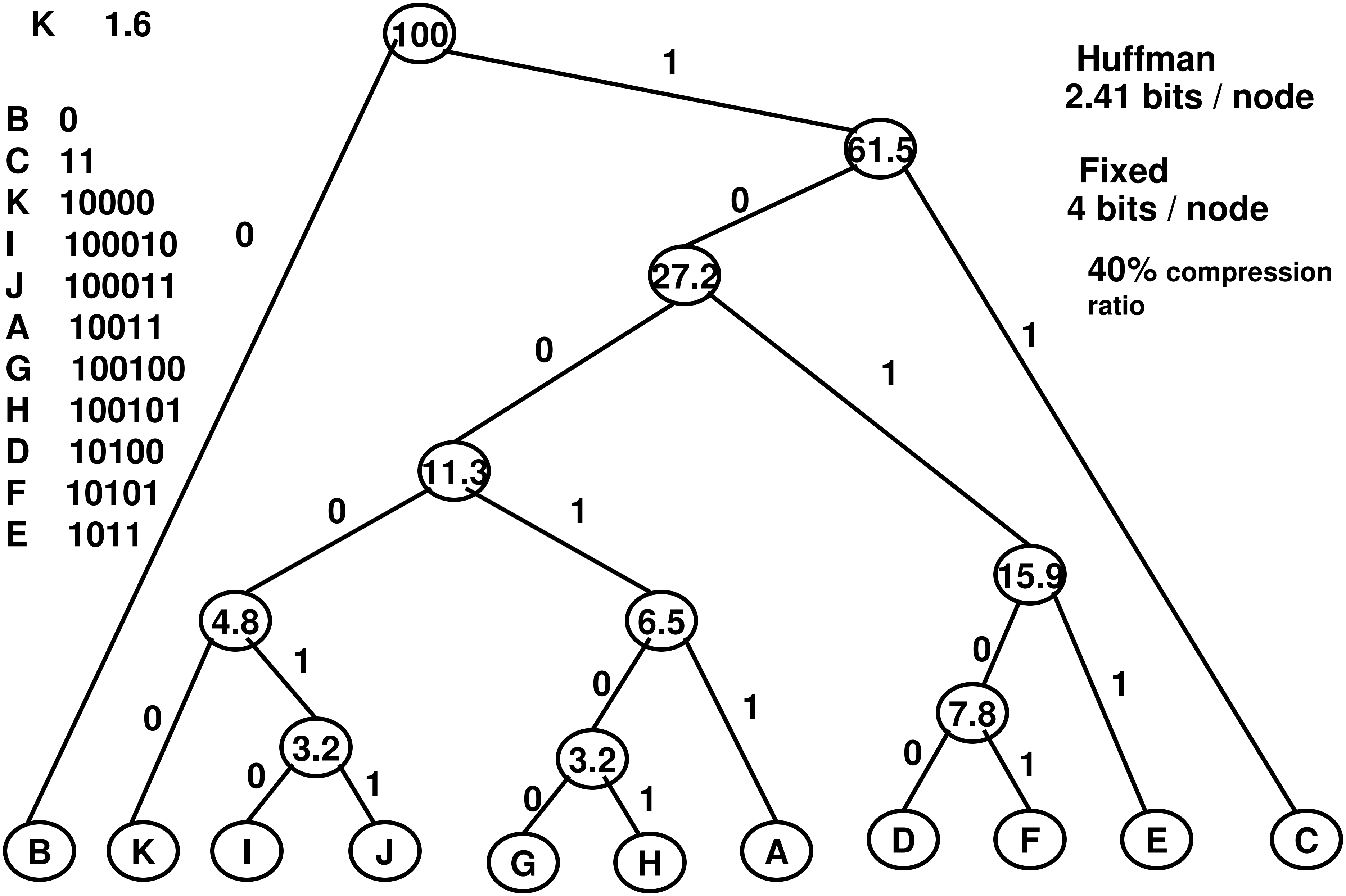
HEBC

100101 1011 0 11

The Huffman codes could be used to efficiently represent paths and frequently used links in the network

A	3.3	B	38.4	C	34.3	D	3.9	E	8.1
F	3.9	G	1.6	H	1.6	I	1.6	J	1.6
K	1.6								

B	0
C	11
K	10000
I	100010
J	100011
A	10011
G	100100
H	100101
D	10100
F	10101
E	1011



Huffman
2.41 bits / node

Fixed
4 bits / node

40% compression ratio

EigenVector Centrality

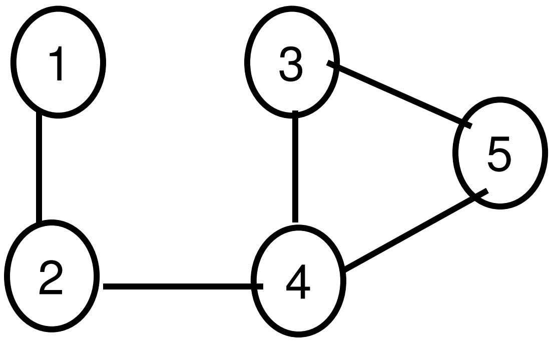
- The idea is to assign scores to all nodes in the network based on the concept that connections to high-scoring nodes contribute more to the score of the node (compared to connections to low-scoring nodes)
- EV Centrality is very useful for analyzing the centrality of nodes in large sparse graphs.
- EV Centrality scores of the vertices are given by the Eigen Vector corresponding to the largest Eigen Value of the Adjacency matrix of a graph.
- The EV Centrality Vector has positive values only
  - Perron-Frobenius Theorem: A real square matrix with positive entries has a unique largest real eigenvalue and that the corresponding Eigenvector has strictly positive components.
- Power Iteration for EV Centrality

Eigen Vector at iteration i+1

$$X_{i+1} = \frac{AX_i}{|| AX_i ||}$$

$|| A X_i ||$ is the normalized value of the product matrix: $A X_i$

EigenVector Centrality Example (1)



$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

Let X0 = $\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$

Iteration 1

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 2 \\ 3 \\ 2 \end{bmatrix} \equiv \begin{bmatrix} 0.213 \\ 0.426 \\ 0.426 \\ 0.639 \\ 0.426 \end{bmatrix}$$

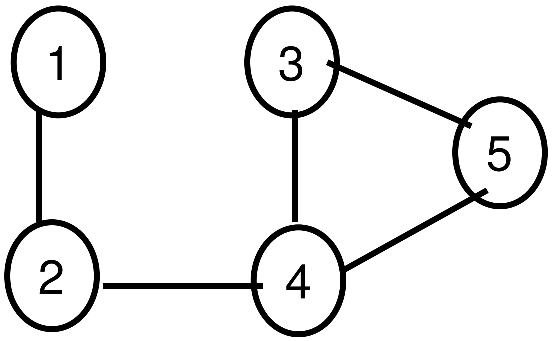
Normalized Value = 4.69

Iteration 2

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0.213 \\ 0.426 \\ 0.426 \\ 0.639 \\ 0.426 \end{bmatrix} = \begin{bmatrix} 0.426 \\ 0.852 \\ 1.065 \\ 1.278 \\ 1.065 \end{bmatrix} \equiv \begin{bmatrix} 0.195 \\ 0.389 \\ 0.486 \\ 0.584 \\ 0.486 \end{bmatrix}$$

Normalized Value = 2.19

EigenVector Centrality Example (1)



$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

Let X0 = $\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$

Iteration 3

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0.195 \\ 0.389 \\ 0.486 \\ 0.584 \\ 0.486 \end{bmatrix} = \begin{bmatrix} 0.389 \\ 0.779 \\ 1.07 \\ 1.361 \\ 1.07 \end{bmatrix} \equiv \begin{bmatrix} 0.176 \\ 0.352 \\ 0.484 \\ 0.616 \\ 0.484 \end{bmatrix}$$

Normalized Value = 2.21

Iteration 4

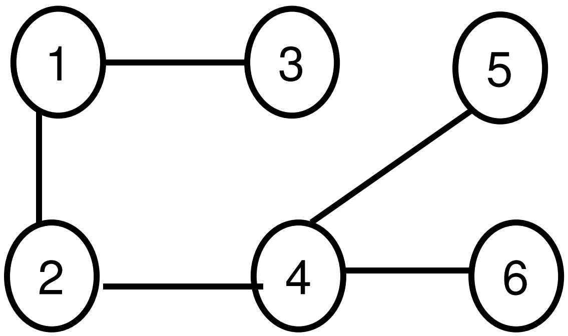
$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0.176 \\ 0.352 \\ 0.484 \\ 0.616 \\ 0.484 \end{bmatrix} = \begin{bmatrix} 0.352 \\ 0.792 \\ 1.100 \\ 1.320 \\ 1.100 \end{bmatrix}$$

Normalized Value = 2.21 converges

Eigen Vector Centrality

$$\begin{bmatrix} 0.176 \\ 0.352 \\ 0.484 \\ 0.616 \\ 0.484 \end{bmatrix}$$

EigenVector Centrality Example (2)



$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Let X0 = $\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$

Iteration 1

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 1 \\ 3 \\ 1 \\ 1 \end{bmatrix} \equiv \begin{bmatrix} 0.447 \\ 0.447 \\ 0.224 \\ 0.671 \\ 0.224 \\ 0.224 \end{bmatrix}$$

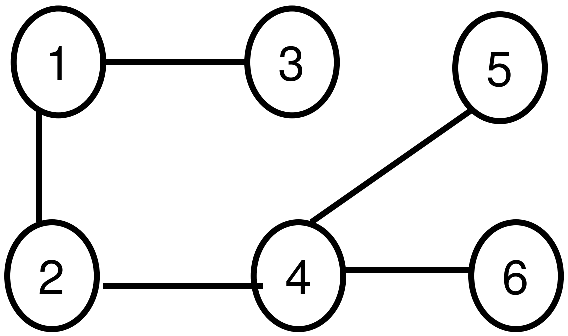
Normalized Value = 4.472

Iteration 2

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.447 \\ 0.447 \\ 0.224 \\ 0.671 \\ 0.224 \\ 0.224 \end{bmatrix} = \begin{bmatrix} 0.671 \\ 0.671 \\ 0.447 \\ 0.895 \\ 0.671 \\ 0.671 \end{bmatrix} \equiv \begin{bmatrix} 0.401 \\ 0.401 \\ 0.267 \\ 0.535 \\ 0.401 \\ 0.401 \end{bmatrix}$$

Normalized Value = 1.674

EigenVector Centrality Example (2)



$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Let X0 = $\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$

<u>Iteration 3</u>

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.401 \\ 0.401 \\ 0.267 \\ 0.535 \\ 0.401 \\ 0.401 \end{bmatrix} = \begin{bmatrix} 0.668 \\ 0.936 \\ 0.401 \\ 1.203 \\ 0.535 \\ 0.535 \end{bmatrix} \equiv \begin{bmatrix} 0.357 \\ 0.500 \\ 0.214 \\ 0.643 \\ 0.286 \\ 0.286 \end{bmatrix}$$

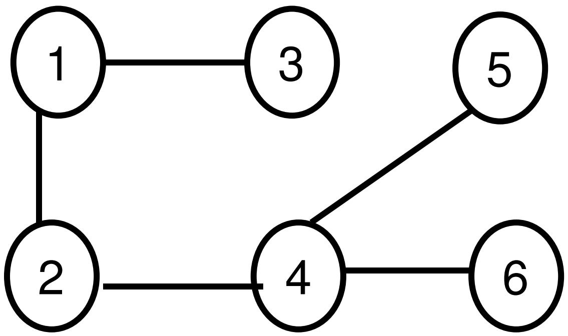
Normalized Value = 1.872

<u>Iteration 4</u>

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.357 \\ 0.500 \\ 0.214 \\ 0.643 \\ 0.286 \\ 0.286 \end{bmatrix} = \begin{bmatrix} 0.714 \\ 1.000 \\ 0.357 \\ 1.072 \\ 0.643 \\ 0.643 \end{bmatrix} \equiv \begin{bmatrix} 0.376 \\ 0.526 \\ 0.188 \\ 0.564 \\ 0.338 \\ 0.338 \end{bmatrix}$$

Normalized Value = 1. 901

EigenVector Centrality Example (2)



Iteration 5

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.376 \\ 0.526 \\ 0.188 \\ 0.564 \\ 0.338 \\ 0.338 \end{bmatrix} = \begin{bmatrix} 0.714 \\ 0.940 \\ 0.376 \\ 1.202 \\ 0.564 \\ 0.564 \end{bmatrix} \equiv \begin{bmatrix} 0.376 \\ 0.494 \\ 0.198 \\ 0.632 \\ 0.297 \\ 0.297 \end{bmatrix}$$

Normalized Value = 1. 901 converges

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Let X0 = $\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$

EigenVector Centrality

$$\begin{bmatrix} 0.376 \\ 0.494 \\ 0.198 \\ 0.632 \\ 0.297 \\ 0.297 \end{bmatrix}$$

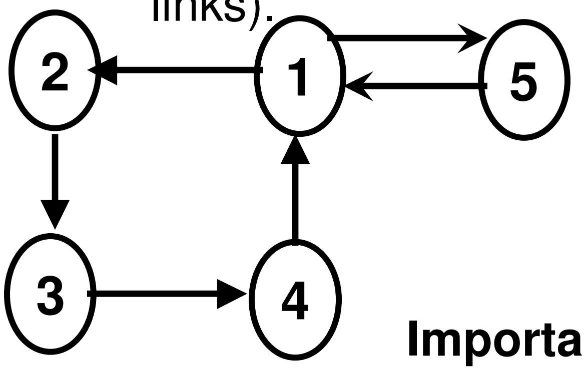
Node Ranking

4
2
1
5
6
3

Note that we typically stop when the EigenVector values converge.
For exam purposes, we will Stop when the Normalized value converges.

Eigen Vector Centrality for Directed Graphs

- For directed graphs, we can use the Eigen Vector centrality to evaluate the "importance" of a node (based on the out-degree Eigen Vector) and the "prestige" of a node (through the in-degree Eigen Vector)
  - A node is considered to be more important if it has out-going links to nodes that in turn have a larger out-degree (i.e., more out-going links).
  - A node is considered to have a higher "prestige", if it has in-coming links from nodes that themselves have a larger in-degree (i.e., more in-coming links).



0	**1**	**0**	**0**	**1**
0	**0**	**1**	**0**	**0**
0	**0**	**0**	**1**	**0**
1	**0**	**0**	**0**	**0**
1	**0**	**0**	**0**	**0**

Out-going links based Adj. Matrix

Importance of Nodes (Out-deg. Centrality)

Node	Score
1	1.272
4	1.0
5	1.0
3	0.786
2	0.618

0	**0**	**0**	**1**	**1**
1	**0**	**0**	**0**	**0**
0	**1**	**0**	**0**	**0**
0	**0**	**1**	**0**	**0**
1	**0**	**0**	**0**	**0**

In-coming links based Adj. Matrix

Prestige of Nodes (In-deg. Centrality)

Node	Score
1	1.272
2	1.0
5	1.0
3	0.786
4	0.618

Katz Centrality

- Katz Centrality computes the relative influence of a node within a network by measuring the number of the immediate neighbors and also all other nodes in the network that connect to the node under consideration through these immediate neighbors.
- Connections made with neighbors at distance d (d - # hops) are penalized by an attenuation factor α^d.
- The attenuation factor α should be lower than the inverse of the largest Eigen Value of the Adjacency matrix.
- Suited specifically for directed acyclic graphs (DAGs) wherein there are no cycles to influence the number of paths of a certain hop count between any two vertices.

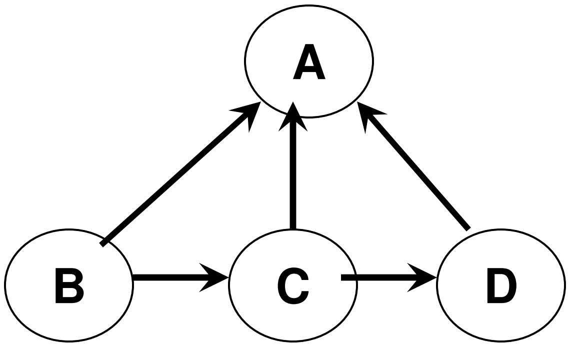
$$C_{Katz}(i) = \sum_{k=1}^{\infty} \alpha^k \sum_{j=1}^{n} \left(A^k\right)_{ji}$$

A – adjacency matrix

A^k captures the number of paths of length (hops) *k* between any two nodes in the graph

α is the attenuation factor

Katz Centrality Example (1)



Adjacency Matrix A = $\begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$

$\alpha = 0.1$

C_{Katz} = [0 0 0 0]

Distance d = 1

$A^1 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$

[3 0 1 1]

$* (0.1)^1$

+

[0 0 0 0]

C_{Katz} = [0.3 0 0.1 0.1]

Distance d = 2

$A^2 = A^1 * A = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$

[2 0 0 1]

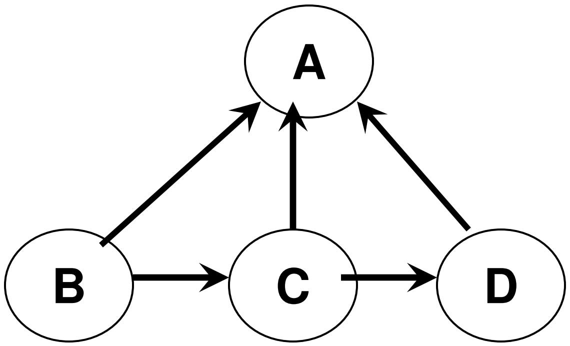
$* (0.1)^2$

+

[0.3 0 0.1 0.1]

C_{Katz} = [0.32 0 0.1 0.11]

Katz Centrality Example (1)



Adjacency Matrix A = $\begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$

C_{Katz} $\begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}$

$\alpha = 0.1$

Distance d = 3

$$A^3 = A^2 * A = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}$

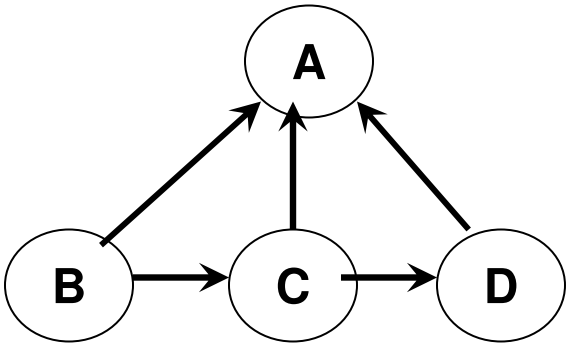
$* (0.1)^3$

$+$

$\begin{bmatrix} 0.32 & 0 & 0.1 & 0.11 \end{bmatrix}$

C_{Katz} $\begin{bmatrix} 0.321 & 0 & 0.1 & 0.11 \end{bmatrix}$

Katz Centrality Example (1)



Adjacency Matrix $A = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$

C_{Katz} $\begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}$

$\alpha = 0.1$

Distance d = 4

$$A^4 = A^3 * A = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

For DAGs, We could stop once all the entries in A^k are 0 (or) if distance equals to Number of Nodes – 1, whichever is larger.

$\begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}$

$* (0.1)^4$

$+$

C_{Katz} $\begin{bmatrix} 0.321 & 0 & 0.1 & 0.11 \end{bmatrix}$

$\begin{bmatrix} 0.321 & 0 & 0.1 & 0.11 \end{bmatrix}$

Final Katz Centrality Vector for the given DAG
[0.321 0 0.1 0.11]

Ranking of the Nodes: A D C B

Subgraph Centrality

- The subgraph centrality of a node is a measure of the number of sub graphs a node is part of.
  - Gives more importance to the smaller sub graphs
  - Measured as the weighted sum of the number of closed walks of particular length (l = 1, 2, 3, ….) that a node is part of. The weights are $1/l!$
  - For a given adjacency matrix A, A^l gives the number of closed walks of length l from a vertex to another vertex (incl. itself).

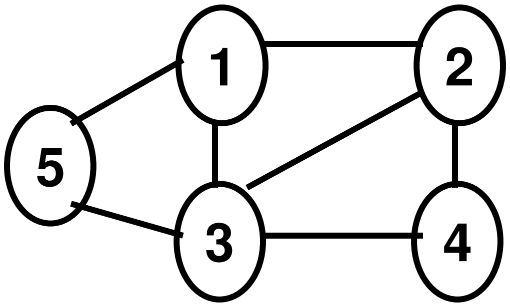
$$EE(i) = \sum_{l=0}^{\infty} \frac{\left(A^l\right)_{ii}}{l!}$$

In closed form

$$EE(i) = \left(e^A\right)_{ii} = \sum_{j=1}^{n} \left[\varphi_j(i)\right]^2 e^{\lambda_j}$$

where $\varphi_j(i)$ is the ith entry of the jth Eigenvector associated with Eigenvalue λ_j

Subgraph Centrality Example (1)



$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Raw results from the website

Real Eigenvalues: { -1.6180339887498933 ; -1.4728339089952571 ; -0.4625984229747743 ; 0.618033988749895 ; 2.9354323319700297 }

Eigenvectors:

for Eigenvalue -1.6180339887498933:

[-1.6180339887499287 ; 1.618033988749923 ; 0 ; -1 ; 1]

for Eigenvalue -1.4728339089952571:

[0.3210368162407646 ; 0.32103681624073577 ; -1.7938707252360133 ; 1 ; 1]

for Eigenvalue -0.4625984229747743:

[-1.1617021380432389 ; -1.1617021380432393 ; 0.6991037150684648 ; 1 ; 1]

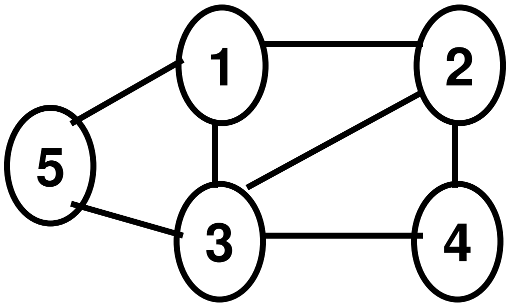
for Eigenvalue 0.618033988749895:

[0.6180339887498949 ; -0.6180339887498948 ; 0 ; -1 ; 1]

for Eigenvalue 2.9354323319700297:

[1.340665321802488 ; 1.3406653218024878 ; 1.5947670101675404 ; 1 ; 1]

Subgraph Centrality Example (2)



A =

0	1	1	0	1
1	0	1	1	0
1	1	0	1	1
0	1	1	0	0
1	0	1	0	0

λ1 = -1.618
λ2 = -1.473
λ3 = -0.463
λ4 = 0.618
λ5 = 2.935

Node IDs

Eigenvalue entries		1	2	3	4	5		$e^{\lambda j}$
	1	-1.618	1.618	0	-1	1	λ1 = -1.618	0.2
	2	0.321	0.321	-1.794	1	1	λ2 = -1.473	0.23
	3	-1.162	-1.162	0.699	1	1	λ3 = -0.463	0.63
	4	0.618	-0.618	0	-1	1	λ4 = 0.618	1.852
	5	1.341	1.341	1.595	1	1	λ5 = 2.935	18.654

EE(Node 1) = { $(-1.618)^2$ * e^(-1.618) + $(0.321)^2$ e^(-1.473) + $(-1.162)^2$ e^(-0.463) + $(0.618)^2$ * e^(0.618) + $(1.341)^2$ * e^(2.935) } = **35.65**

EE(Node 2) = { $(1.618)^2$ * e^(-1.618) + $(0.321)^2$ e^(-1.473) + $(-1.162)^2$ e^(-0.463) + $(-0.618)^2$ * e^(0.618) + $(1.341)^2$ * e^(2.935) } = **35.65**

Node IDs

	1	2	3	4	5
1	-1.618	1.618	0	-1	1
2	0.321	0.321	-1.794	1	1
3	-1.162	-1.162	0.699	1	1
4	0.618	-0.618	0	-1	1
5	1.341	1.341	1.595	1	1

λ1 = -1.618	**0.2**
λ2 = -1.473	**0.23**
λ3 = -0.463	**0.63**
λ4 = 0.618	**1.852**
λ5 = 2.935	**18.654**

EE(Node 3) = $\{ (0)^2 * e^\wedge(-1.618) + (-1.794)^2\, e^\wedge(-1.473) + (0.699)^2\, e^\wedge(-0.463) + (0)^2 * e^\wedge(0.618) + (1.595)^2 * e^\wedge(2.935) \} =$ **48.5**

EE(Node 4) = $\{ (-1)^2 * e^\wedge(-1.618) + (1)^2\, e^\wedge(-1.473) + (1)^2\, e^\wedge(-0.463) + (-1)^2 * e^\wedge(0.618) + (1)^2 * e^\wedge(2.935) \} =$ **21.6**

EE(Node 5) = $\{ (1)^2 * e^\wedge(-1.618) + (1)^2\, e^\wedge(-1.473) + (1)^2\, e^\wedge(-0.463) + (1)^2 * e^\wedge(0.618) + (1)^2 * e^\wedge(2.935) \} =$ **21.6**